

ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»
ІНСТИТУТ ЕКОНОМІКИ, УПРАВЛІННЯ ТА ІНФОРМАЦІЙНИХ
ТЕХНОЛОГІЙ
ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ
ІНФОРМАТИКИ

Допускається до захисту

Завідувач кафедри _____ О.О. Ємець
«_____» _____ 2019 р.

**ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО ДИПЛОМНОЇ РОБОТИ**

на тему

**ТРЕНАЖЕР З ТЕМИ «АЛГЕБРА ВИСЛОВЛЮВАНЬ» ДИСТАНЦІЙНОГО
НАВЧАЛЬНОГО КУРСУ «МАТЕМАТИЧНА ЛОГІКА» ТА РОЗРОБКА ЙОГО
ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

зі спеціальності «Комп'ютерні науки»

Виконавець роботи: студент групи КН-61 Долгов Владислав Олегович

_____ «__» _____ 2019 р.

Науковий керівник: к.ф.-м.н., доц., Черненко Оксана Олексіївна

_____ «__» _____ 2019 р.

Полтава-2019

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ.....	4
ВСТУП	5
1 ПОСТАНОВКА ЗАДАЧ.....	7
1.1 Основні вимоги до змісту роботи.....	7
1.2 Постановка задач для реалізації програми-тренажера з теми «Алгебра висловлювань».....	7
2 ІНФОРМАЦІЙНИЙ ОГЛЯД.....	9
2.1 Огляд існуючих розробок.....	9
2.2 Переваги розглянутих робіт.....	11
2.3 Недоліки розглянутих робіт.....	12
2.4 Висновки за розділом 2.....	12
3 ТЕОРЕТИЧНА ЧАСТИНА.....	13
3.1 Алгоритмізація роботи тренажера.....	13
3.1.1 Основні теоретичні відомості з теми «Алгебра висловлювань».....	13
3.1.2 Приклади та розв'язки завдань з теми «Алгебра висловлювань».....	19
3.1.3 Алгоритм роботи тренажера.....	23
3.3 Розробка блок-схеми алгоритму роботи навчального тренажера.....	34
3.3 Обґрунтування вибору програмних засобів	37
3.4 Висновки за розділом 3.....	38
4 ПРАКТИЧНА ЧАСТИНА.....	39
4.1 Опис процесу програмної реалізації.....	39
4.2 Опис програми.....	41
4.3 Тестування коректності роботи навчального тренажера.....	50
4.4 Інструкція для користувачів програми.....	68
4.5 Висновки за розділом 4.....	76

ВИСНОВКИ.....	77
СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ.....	78
ДОДАТОК А. ЛІСТИНГ ПРОГРАМИ НА МОВІ PYTHON.....	80

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

Умовне скорочення	Пояснення умовного скорочення
ВНЗ	Вищий навчальний заклад
ДН	Дистанційне навчання
GUI	Графічний інтерфейс користувача
Moodle	Modular Object-Oriented Dynamic Learning Environment

ВСТУП

Дистанційне навчання – це можливість здобування освіти без необхідності перебування в навчальному закладі. Ця система дуже актуальна для студентів, які живуть далеко від навчального закладу, працюють, або просто не мають змоги відвідувати лекції. Студенту для кращого сприйняття матеріалу, необхідна допомога викладача. Але викладач не завжди може бути на зв'язку зі студентом, а студент не завжди може зрозуміти викладений матеріал. Найкращий шлях подолання цієї проблеми – створення спеціального навчального тренажера. Завдяки інтерактивній формі взаємодії між студентом та навчальною програмою, тренажер здатен частково замінити викладача, а студент, в свою чергу, буде краще сприймати викладений матеріал та розуміти його сутність.

Також було виявлено, що серед безлічі дистанційних курсів та інтерактивних додатків, не зустрічається навчальний тренажер, який міг би допомогти в вивченні матеріалу з теми «Алгебра висловлювань» по дисципліні «Математична логіка». Саме тому його створення та впровадження в систему дистанційного навчання «Полтавського університету економіки і торгівлі» є досить актуальною. Також це допоможе спростити студентам самостійне вивчення матеріалу, а керівнику лише залишиться перевірити рівень знань студентів, за допомогою іспитів та спеціальних завдань.

Мета дипломного проекту – розробка програмного забезпечення для тренажера з теми «Алгебра висловлювань» дистанційного навчального курсу «Математична логіка».

Об'єкт розробки – програмне забезпечення дистанційного навчання.

Предмет розробки – додаток у вигляді програми з теми «Алгебра висловлювань», який буде інтегровано в дистанційний навчальний курс «Математична логіка».

Методи розробки – мова розробки Java, методи математичної логіки.

Дипломна робота складається з чотирьох розділів, а саме: постановка задачі, інформаційний огляд, теоретична частина. Структура роботи побудована так, що дає змогу логічного представлення матеріалу та розкриття теми роботи.

Для вирішення поставленої мети потрібно вирішити наступні задачі:

- проаналізувати існуючі розробки, в яких розглянуто подібні до теми питання;
- розробити алгоритм роботи тренажеру та його блок-схему;
- реалізувати тренажер в програмному вигляді, протестувати його працездатність та написати інструкцію з його використання.

Для вирішення поставлених в роботі задач було:

- проаналізовано переваги та недоліки існуючих тренажерів;
- розроблено алгоритм роботи тренажера;
- програмна реалізація алгоритму тренажера.

Магістерська робота складається з вступу, чотирьох розділів (постановка задач, інформаційний огляд, теоретична частина, практична частина), списку використаних джерел і додатків.

Результатом виконання дипломної роботи є створення програмного забезпечення для тренажера з теми «Алгебра висловлення» за допомогою мови програмування *Java*. Практичне значення одержаних результатів полягає в можливості використання розробленого програмного забезпечення при вивченні зазначеної теми.

1 ПОСТАНОВКА ЗАДАЧ

1.1 Основні вимоги до змісту роботи

При оформленні матеріалів дипломної роботи потрібно дотримуватись методичних рекомендацій щодо оформлення роботи студента зі спеціальності «Комп'ютерні науки». Зміст роботи повинен відповідати завданню затвердженому керівником та завідувачем кафедри[1].

Викладені матеріали повинні відповідати таким пунктам:

- чіткість, логічність та послідовність викладення матеріалу;
- переконливість матеріалу;
- чіткість та точність формулювань, що допомагають уникнути неоднозначних висновків при тлумаченні результатів;
- обґрунтування рекомендацій та пропозицій.

Робота повинна містити:

- актуальність теми дослідження або розробленого проекту;
- обґрунтування вибору напрямку дослідження, методи розв'язку задач, а також їх порівняльна оцінка;
- аналіз та узагальнення результатів;
- опис дії та характеристики розроблених програм;
- наявність блок-схеми програми;
- використання контрольних прикладів;
- англomовний варіант тренажера;
- оцінка розв'язку поставленої задачі, наукова та практична цінність роботи.

1.2 Постановка задачі для реалізації програми–тренажера з теми «Алгебра висловлювань»

Головним завданням магістерського дипломного проекту є розробка програмного забезпечення для тренажера з теми «Алгебра висловлювань»

дистанційного навчального курсу «Математична логіка». З метою підвищити ефективність проекту, планується зробити додаток сумісним з системою дистанційного навчання Moodle.

Виходячи з викладеного вище, щоб досягнути поставленої мети потрібно виконати наступні завдання:

- зробити аналіз робіт, в яких розглянуто аналогічні до теми розробки питання;
- розробити алгоритму тренажера та його блок-схему;
- обґрунтувати вибір середовища розробки для реалізації тренажера;
- програмно реалізувати тренажер, протестувати його працездатність його роботи та написати інструкцію по його використанню.

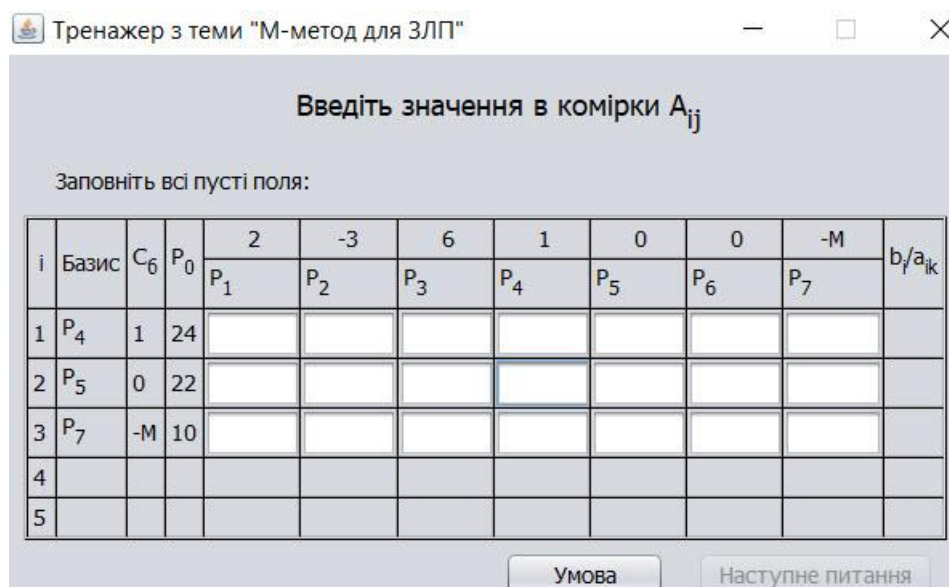
Додаток розробляється з метою полегшити студентам процес вивчення та засвоєння теоретичного матеріалу. Тренажер повинен мати не тільки тестові завдання та перевірку знань студента, але й мати підказки у вигляді витягів з лекції. Також розроблений тренажер повинен мати зрозумілий інтерфейс, щоб його було зручно використовувати, зосередившись на навчальному процесі.

2 ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Огляд існуючих розробок

На наш час існує багато навчальних програм-тренажерів розроблених у тій чи іншій темі та напрямку підготовки. Тренажеру, який стосувався теми «Алгебра висловлювань» не було виявлено, проте було розглянуто близькі до цієї теми тренажери.

Серед них варто виділити тренажер «М-метод для ЗЛП»[2], в якому зображено приклад покрокового розв'язування ЗЛП (зображено на рисунку 1). В разі помилки студенту дається підказка, яка дозволить зрозуміти суть помилки та виправити її.



i	Базис	C_6	P_0	2	-3	6	1	0	0	-M	b/a_{ik}
				P_1	P_2	P_3	P_4	P_5	P_6	P_7	
1	P_4	1	24								
2	P_5	0	22								
3	P_7	-M	10								
4											
5											

Рисунок 2.1 – Заповнення комірок симплекс методу у тренажері «М-метод для ЗЛП»

Наступний приклад, який варто зазначити, – це тренажер з теми «Метод гілок та меж в задачі про найкоротший шлях»[3]. В тренажері на вибір надається декілька прикладів для розв'язку. Обирається будь-який з них та в тестовому режимі «крок за кроком» розв'язуємо задачу (приклад зображено на рисунку 2). У разі помилки демонструється витяг з лекції стосовно того етапу, на якому знаходиться користувач.

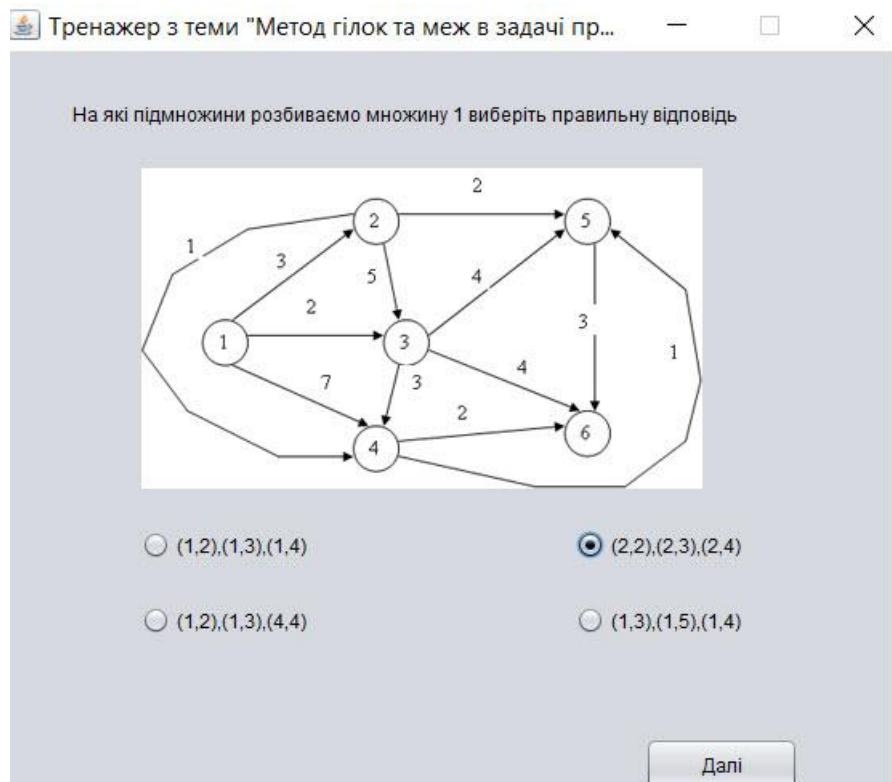


Рисунок 2.2 – Приклад задачі в тренажері з теми «Метод гілок та меж в задачі про найкоротший шлях»

Також вартий уваги тренажер з теми «Графічний метод розв'язання ЗЛП»[4]. В цьому тренажері представлений набір завдань зростаючої складності (приклад одного завдання зображено на рисунку 3). В будь який момент користувач може вибрати будь-яке завдання на вибір, а в разі помилки отримає підказку щодо правильної відповіді.

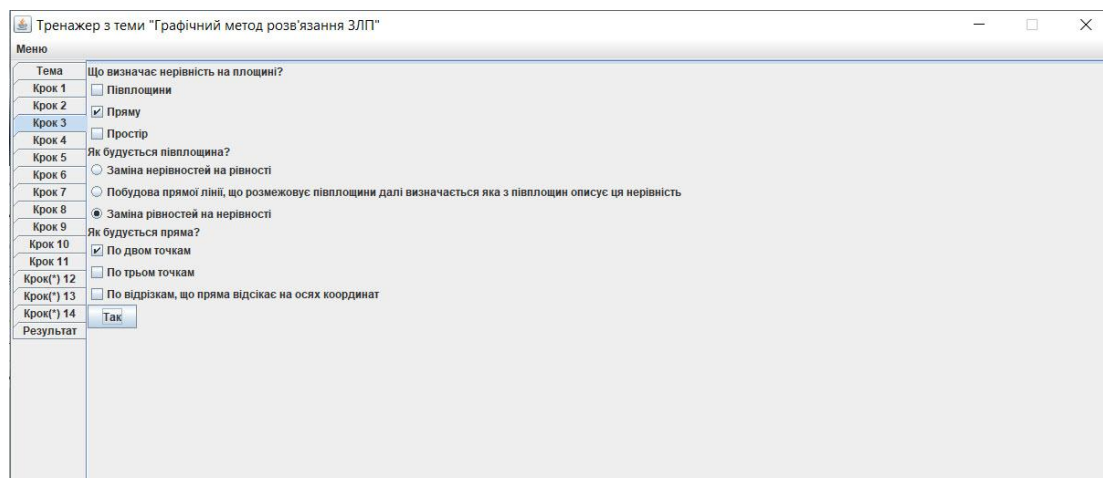


Рисунок 2.3 – Одне з завдань тренажеру «Графічний метод розв'язування ЗЛП»

Серед найближчих тренажерів слід виділити тренажер з теми «Нормальні алгоритми»[5]. Тренажер дає змогу у режимі «крок за кроком» розв'язати завдання, та в разі помилки сповістити про це студента. Тренажер надає можливість вибору з декількох завдань. Користувач проходить завдання в тестовому режимі. У разі правильної відповіді користувач переходить до наступного кроку, а у разі помилки отримує сповіщення про неправильність розв'язку та підказку.

Також близьким тренажером є тренажер з теми «Мови і граматики» з навчального курсу «Теорія програмування»[6]. В тренажері є три завдання, для вирішення кожного з них користувачеві потрібно пройти декілька кроків. Кожен крок розроблений у вигляді запитання з декількома варіантами відповіді. Якщо користувач відповідає правильно, то він переходить до наступного кроку, а в разі неправильної відповіді на екрані з'являється підказка і перехід до наступного запитання. Також перед початком є можливість переглянути теоретичний матеріал.

2.2 Переваги розглянутих робіт

На основі розглянутих тренажерів було сформовано наступні критерії, які будуть враховуватися під час розробки тренажера. А саме під час огляду були виявлені такі позитивні рішення, які слід розвивати далі, а саме:

- різноманітність завдань;
- сортування матеріалу в порядку складності;
- наявність витягів з лекцій;
- покроковий розв'язок прикладів за допомогою яких легко зрозуміти принцип та хід дій виконання завдання;
- навички які допоможуть розв'язати ЗЛП самостійно;
- наявність підказок, які допомагають розв'язати задачу;
- зручний покроковий шлях розв'язку задач;

- витяги з лекції, у разі неправильної відповіді;
- можливість обрати приклад.

2.3 Недоліки розглянутих робіт

Також під час огляду робіт були виявлені наступні вади, яких варто уникати, а саме:

- не вистачає уривків з лекції, які допомогли зрозуміти тему в цілому.
- в деяких роботах замало прикладів;
- в деяких роботах із завдань тільки тести;
- занадто прості підказки, які відразу показують правильну відповідь.

2.4 Висновки за розділом 2

У результаті перегляду та аналізу програмних аналогів було зроблено наступні висновки:

1. При розробці тренажерів використовуються різні середовища розробки, вибір яких залежить від поставленої мети.
2. Основним критерієм розроблених тренажерів, є можливість впровадити їх до системи дистанційного навчання навчального закладу, та сумісність з різними операційними системами, щоб студенти могли мати до них доступ.
3. Тренажери найчастіше будуються у вигляді тестових завдань, коли користувачу потрібно відповісти на запитання, обравши одну з запропонованих відповідей.
4. Під час аналізу розроблених робіт було виявлено безліч навчальних тренажерів на різні теми, але не було помічено жодного тренажера з теми «Алгебра висловлювань», що засвідчує необхідність та актуальність цієї теми.

3 ТЕОРЕТИЧНА ЧАСТИНА

3.1 Алгоритмізація роботи тренажера

Перед тим, як навести алгоритм роботи тренажера, потрібно розглянути основні теоретичні відомості з теми «Алгебра висловлювань», а також розглянути послідовність розв'язання прикладів, які покладені в основу його роботи.

3.1.1 Теоретичні відомості з теми «Алгебра висловлювань»

Алгебра висловлювань – це розділ математичної логіки, де досліджується операції на висловлюваннях. Висловлення – це твердження, яке може приймати форму тільки істинного або тільки хибного значення, і в жодному разі не те чи інше одночасно. Тобто вони виявляються бінарними, або, як їх ще називають, двійковими висловленнями. Наприклад висловлювання «Яблуко – це фрукт» це істинним, а висловлення «Груша – це овоч» є хибним. Також висловлення повинне бути у вигляді оповідального речення. Наприклад речення «Ворскла – це річка» є істинним висловленням, а речення «Яка це річка?» або «Довжина Ворскли більше 400 км» не є висловлення, адже в першому випадку воно не оповідальне, а в другому не точне. Отже маємо висновок, що висловлення – це оповідальне речення, про яке можна сказати, що воно є істинним або хибним, але не те й інше одночасно. З точки зору природної мови воно не є повним, але воно необхідне для використання логіки висловлень. В свою чергу істинність або хибність висловлення, називається істинним значенням цього висловлення [7].

Слід також зазначити, що висловлення в свою чергу діляться на прості та складні. Складні висловлення складаються з декількох простих висловлень(вони також називаються «атомами»), поєднаних сполучниками. В свою чергу прості висловлювання неможливо поділити на ще більш простіші. Прості висловлення найчастіше позначають латинськими літерами алфавіту, значення істинності позначають цифрами 1 (істина) та 0 (хибність), або латинськими T(True – в перекладі з англійської «правда») та F(False – в перекладі з англійської «брехня»).

При складанні логічних зв'язків між висловлюваннями використовуються спеціальні символи, а саме:

\neg – заперечення (логічне «ні») істинне тому випадку коли висловлення хибне;

\wedge – кон'юнкція (множення, логічне «і») істинна тоді, коли істинні обидва висловлення;

\vee – диз'юнкція (додавання, логічне «або») хибна в тому випадку, коли хибні обидва висловлення;

\rightarrow – імплікація (логічне «якщо, ... то ...») хибна коли перше висловлення істинне, а друге хибне;

\sim – еквівалентність (рівнозначність, логічне «тоді і тільки тоді, коли») істинне лише в тому випадку, коли обидва висловлення істинні або обидва висловлення хибні.

Такі операції як \neg , \wedge , \vee , \rightarrow є бінарними логічними зв'язками які зв'язують між собою два логічних висловлення, а \neg є унарною операцією яка застосовується до одного висловлення. Також кожна операція має свій пріоритет в такій послідовності: \neg , \wedge , \vee , \rightarrow , \sim .

В математичній логіці правильно побудована формула визначається послідовно за такими правилами:

1. Атом є формулою.
2. Якщо A і B – формули, то $(A \sim B)$, $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$ і $\neg A$ – також формули.
3. Ніяких формул, крім створених вище правилами, не існує[8].

У таблиці 3.1 зображено значення істинності висловлень, де A і B це висловлення, 1 – істина, а 0 – хибність.

Таблиця 3.1– Значення істинності висловлень

A	B	$\neg A$	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \sim B$
0	0	1	0	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	0	0
1	1	0	1	1	1	1

Виходячи з таблиці, якщо значення істинності простих висловлень(атомів) відомі, то значення істинності складних висловлень може бути визначено за допомогою наведеної вище таблиці.

Інтерпретування формули – це присвоєння їй одного з двох значень істинності. Набір правил інтерпретації формул повинен бути композиційним, тобто значення формули має бути функцією значень її складових. Для інтерпретації формул можна використовувати таблиці істинності[9].

Наприклад, для демонстрування істинності формул $A \rightarrow B \sim \neg A \vee B$ за будь-яких інтерпретацій зображена таблиця.

Таблиця 3.2 – Інтерпретація формули $A \rightarrow B \sim \neg A \vee B$

A	B	$A \rightarrow B$	$\neg A$	$\neg A \vee B$	$A \rightarrow B \sim \neg A \vee B$
0	0	1	1	1	1
0	1	1	1	1	1
1	0	0	0	0	1
1	1	1	0	1	1

Формули розподіляються на три види, в залежності від їх інтерпретацій, а саме: тавтологічні, протиріччі та нейтральні. Варіанти інтерпретацій бувають двох видів: здійсненні (якщо є хоча б одна інтерпретація за якої формула набуває істинного значення) та спростована (коли існує хоча б одна інтерпретація за якої формула набуває хибного значення). Тавтологію позначають так: $\models A$.

Тавтологія (також називають тотожно-істинною або загальнозначущою) – це формула, яка за будь-яких інтерпретацій змінних набуває істинного значення.

Протиріччя (тотожно-хибна) – формула, яка за будь-яких інтерпретацій її складових набуває хибного значення.

Нейтральна (несуперечлива або незначуща) – це формула яка сприймає на одних інтерпретаціях приймає значення істина, а на других «Хибність».

Результати вище описаних означень:

- формула A є тавтологією лише тоді, коли A не є спростовною;
- формула A є протиріччям лише тоді, коли A не здійсненою;
- формула A є тавтологією лише тоді, коли $\neg A$ є протиріччям;
- формула A є протиріччям лише тоді, коли $\neg A$ тавтологія;
- формула $A \sim B$ – тавтологія лише тоді, коли A та B рівносильні (набувають однакових значень) за всіма наборами значень змінних.

Отже, дві формули рівносильні, лише у тому випадку, коли за будь-якої інтерпретації їх змінних вони мають однакові значення (позначаються: $A = B$).

Дві формули A та B рівносильні лише в тому випадку, коли $| = A \sim B$.

Наприклад, нехай P_1, P_2, \dots, P_m це сукупність простих компонентів в A та B . Якщо цим компонентам було надано деякі істинні значення, то перша частина розрахунків значень $A \sim B$ буде істинним тільки тоді, коли значення для A та B будуть однакові (рівносильні).

Отже, нехай F_1 – формула, в котрій знаходяться формули A , і нехай F_2 – є результатом заміни цього входження формули A на формулу B .

Тоді:

якщо $| = A \sim B$, то $| = F_1 \sim F_2$;

якщо $| = A \sim B$ і $| = F_1$, то $| = F_2$.

З теореми відомо якщо $| = A$ і $| = A \rightarrow B$, то $| = B$.

Наприклад, P_1, P_2, \dots, P_m є сукупність простих елементів в A та B . Якщо цим компонентам надано певних значень істинності, то перша частина обчислення значень формули $A \rightarrow B$ полягає в обчисленні значень A та B , після чого розрахунки закінчуються використанням таблиці істинності для імплікацій.

Якщо $| = A$ та $| = A \rightarrow B$ впливає, що значення A та $A \rightarrow B$ приймуть істинне значення.

З таблиці для $A \rightarrow B$ виходить, що B теж істинне. Через те що воно має місце для кожних значень компонентів P_1, P_2, \dots, P_m , формула B є загальнозначуща.

Завдяки наданим теоремам, з'являється можливість здійснювати еквівалентні перетворювання формул і здобуття нові загальнозначущі формули.

Наприклад, можна отримати тавтологію з рівносильною заміною знаку $=$ на знак \sim . З рівносильної $A \vee AB = A$ отримаємо тавтологію $| = A \vee AB \sim A$. Для прикладу доведемо тавтологію $| = (A \rightarrow B)(A \rightarrow C) \sim (A \rightarrow BC)$ за допомогою перетворень:

$$(A \rightarrow B)(A \rightarrow C) = (\neg A \vee B)(\neg A \vee C) = \neg A \vee \neg AB \vee \neg AC \vee BC = \neg A \vee BC = A \rightarrow BC.$$

Законами алгебри висловлень називають тотожно-істинні формули та формули рівносильності. Існує безліч законів алгебри висловлень, адже ці закони формуються з множини тавтологій та рівносильностей. Закони які найчастіше зустрічаються в роботі наведено нижче:

$x \wedge 0 = 0, x \wedge 1 = x, x \vee 0 = x, x \vee 1 = 1$	– закон сталих;
$x = x$	– закон тотожності;
$\neg \neg x = x$	– закон подвійного заперечення;
$x \wedge \neg x = 0$	– закон протиріччя;
$x \vee \neg x = 1$	– закон вилучення третього;
$x \vee y = y \vee x, x \wedge y = y \wedge x$	– комутативність \vee та \wedge ;
$x \wedge (y \vee z) = x \wedge y \vee x \wedge z$ $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$	– асоціативність \vee та \wedge ;
$x \wedge z$	– перший дистрибутивний закон;
$x \vee y \wedge z = (x \vee y) \wedge (x \vee z) (x \vee z)$	– другий дистрибутивний закон;
$x \wedge (x \vee y) = x$	– перший закон поглинання;
$x \vee x \wedge y = x$	– другий закон поглинання;
$x \vee x = x, x \wedge x = x$	– ідемпотентність;
$x \wedge y \vee x \wedge \neg y = x$	– перший закон склеювання;

$(x \vee y) \wedge (x \vee \neg y) = x$	– другий закон склеювання;
$\neg(x \vee y) = \neg x \wedge \neg y, \neg(x \wedge y) = \neg x \vee \neg y$	– правило де Моргана;
$ = (x \rightarrow y) \wedge x \rightarrow y$	– правило твердження;
$ = (x \rightarrow y) \wedge \neg y \rightarrow \neg x$	– правило спростування.

Важливою характеристикою логічного висновку є співвідношення сумісності між висновком і засновком. Правила висновку використовуються, щоб виводити одні істинні речення з інших істинних речень.

Висловлення B є логічним наслідком висловлення A (позначається $A| = B$), B є істинне на всіх наборах значень змінних, для яких A є істинна. З цього визначення може бути узагальнено на випадок довільного числа засновків таким чином: висловлення B називається логічним наслідком висловлень A_1, A_2, \dots, A_n , якщо $A_1 \wedge A_2 \wedge \dots \wedge A_n \rightarrow B$ – тотожно істинна формула. Наприклад, формула $B = x \vee x$ є логічним наслідком формули: $A = x \wedge (y \vee \neg y)$, тобто $x \wedge (y \vee \neg y)| = x \vee x$.

Формула алгебри висловлень B є логічним наслідком з формули A тільки тоді, коли формула $A \rightarrow B$ є загальнозначущою, тобто $A| = B \sim | = A \rightarrow B$.

Відповідно до визначення імплікації, вираз $A \rightarrow B$ є хибним у разі істинності A та хибності B , а отже, якщо $A \rightarrow B$ це тавтологія, то з істинності A завжди виходить істинність B , тобто $A| = B$.

І, навпаки, якщо $A| = B$, то виключається випадок, коли A є істинне та B – хибне, то $A \rightarrow B$ є істинне на всіх наборах значень змінних, тобто $| = A \rightarrow B$.

Логічний наслідок можна узагальнити на сукупності формул: формула алгебри висловлень B є логічним наслідком формул A_1, A_2, \dots, A_n та позначається як $A_1, A_2, \dots, A_n| = B$, якщо для довільного набору значень з істинності всіх A_i , $i = \overline{1, n}$, на цьому наборі впливає істинність B .

Також якщо формула алгебри висловлень B є логічним наслідком формул A_1, A_2, \dots, A_n тоді й лише тоді, коли формула $A_1 \wedge A_2 \wedge \dots \wedge A_n \rightarrow B$ є загальнозначущою, тобто $A_1, A_2, \dots, A_n| = B \sim | = A_1 \wedge A_2 \wedge \dots \wedge A_n \rightarrow B$.

Для того щоб перевірити правильність логічних твердження, не враховуючи на конкретний зміст висловлень, можна застосувати формули алгебри висловлень.

Що ж стосується «здорового глузду», то він має виявлятися при використанні законів логіки висловлень у її конкретних додатках. Наприклад, висловлення « $A = 100 < 10$ » – хибне. Але якщо число 100 записане у двійковій системі числення, а 10 – у десятковій, то воно стає істинним.

Наприклад, потрібно перевірити правильність наступного твердження. Якщо змінити мікросхему (A), телевізор працюватиме (B), якщо увімкнемо напругу (C). Мікросхему змінили, а напруга вимкнена. Отже, телевізор не працює.

Вище описане твердження можна записати у вигляді: $A \rightarrow B \wedge C$, $A, \neg C \mid = \neg B$.

Оскільки формули A , B , C не містять підформул, то можна перейти до відповідних підформул, то можна перейти до відповідних змінних x , y , z . Тоді логічний висновок буде мати більш зручний вигляд: $x \rightarrow y \wedge z, x, \neg z \mid = \neg y$.

Твердження буде слухним, якщо формула $(x \rightarrow y \wedge z) \wedge x \wedge \neg z \rightarrow \neg y$ є загальнозначущою. При викладках скористаємося основними законами алгебри висловлень: $(x \rightarrow y \wedge z) \wedge x \wedge \neg z \rightarrow \neg y = \neg((\neg x \vee y \wedge z) \wedge x \wedge \neg z) \vee \neg y = \neg 0 \vee \neg y = 1 \vee \neg y = 1$. Формула є загальнозначущою, отже твердження правильне.

Або ще такий приклад. Я піду на лекцію (x) або залишуся в барі й вип'ю кави (y). Я не піду на лекцію. Отже, я залишуся й вип'ю кави.

Маємо логічне слідування: $x \vee y, \neg x \mid = y$. Перевіримо його загальнозначимість: $(x \vee y) \wedge \neg x \rightarrow y = \neg((x \vee y) \wedge x) \vee y = \neg x \wedge \neg y \vee x \vee y = 1 \vee x = 1$. Формула є загальнозначущою, отже твердження правильне.

3.1.2 Приклади та розв'язки завдань з теми «Алгебра висловлювань»

Приклад 1.

Які з наведених виразів є висловленнями? Якщо вираз є висловленням, то вказати яким саме – істинним чи хибним.

- 1) Кожне дійсне число задовольняє нерівність $x^2 \geq 0$.

- 2) Число 168 кратно 9.
- 3) Хай живе математика!
- 4) 15 кратно 3, але не кратно 4.
- 5) Чи існує дійсне число, більше за 3 і менше від $\log_2 9$?
- 6) Ця задача легка.
- 7) Існує найбільше просте число.
- 8) Чи правильна велика теорема Ферма?
- 9) Рівняння $x^2 + 7x + 1 = 0$ має хоч один дійсний корінь.
- 10) Розв'язати рівняння $x^2 + 7x + 1 = 0$.
- 11) Розкрийте підручник на сторінці 23.
- 12) Вчитель сказав: «Розкрийте підручник на сторінці 23».
- 13) Якщо $3 < 2$, то $3^2 < 2^2$.

Розв'язання.

Серед наведених прикладів висловленнями є вирази: 1, 2, 4, 5, 6, 7, 8, 9, 13. Серед них істинними є вирази: 1, 4, 5, 8, 9, 13. Хибні вирази: 2, 7. Також є вирази які можуть бути як істинними, так і хибними: 6, 12. Серед цих прикладів можна виділити прості вирази: 1, 2, 6, 7, 8, 9, 12. Вирази 4, 5, 13 є складеними. Вирази 3, 10, 11 не висловлювання.

Завдання 2.

Яке заперечення висловлювання «Сьогодні середа»? Знати кон'юнкцію висловлювань «Сьогодні п'ятниця» і «Сьогодні падає дощ».

Розв'язання.

Запереченням висловлювання «Сьогодні середа» є висловлювання «Сьогодні не середа». Також слід враховувати що речення, пов'язані з часовою змінною, – не висловлювання доти, доки не визначено момент часу. Кон'юнкція висловлювань «Сьогодні п'ятниця» і «Сьогодні падає дощ» є висловлювання «Сьогодні п'ятниця, і сьогодні падає дощ». Воно буде істинне тільки в дощову п'ятницю, і хибне в будь який інший день або в не дощову п'ятницю.

Завдання 3.

Знайти значення істинності формул алгебри висловлень $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$, $((p \rightarrow q) \wedge p) \rightarrow q$, $(p \rightarrow q) \wedge (p \wedge \bar{q})$ у всіх їх інтерпретаціях.

Розв'язання.

Рівняння формули $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$ має три атоми: p , q , й r . Кожному з цих атомів можна надати \bar{r} ва значення: Т (True) або F (False). Отже ми маємо 8 інтерпретацій. Щоб знайти значення істинності, будуємо таблицю істинності формули $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$ у всіх її ітераціях.

Таблиця 3.3 – Таблиця значення істинності формули $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$

p	q	r	\bar{r}	$(p \wedge q)$	$(p \leftrightarrow \bar{r})$	$(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$
T	T	T	F	T	F	F
T	T	F	T	T	T	T
T	F	T	F	F	F	T
T	F	F	T	F	T	T
F	T	T	F	F	T	T
F	T	F	T	F	F	T
F	F	T	F	F	T	T
F	F	F	T	F	F	T

Отже формула $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$ істинна в 7 із 8 інтерпретаціях.

Рівняння формули $((p \rightarrow q) \wedge p) \rightarrow q$ має два атоми: p та q . В результаті ми маємо 4 ітерації. Будуємо таблицю істинності.

Таблиця 3.4 – Таблиця значення істинності формули $((p \rightarrow q) \wedge p) \rightarrow q$

p	q	$p \rightarrow q$	$(p \rightarrow q) \wedge p$	$((p \rightarrow q) \wedge p) \rightarrow q$
T	T	T	T	T
T	F	F	F	T
F	T	F	F	T
F	F	F	F	T

Отже формула $((p \rightarrow q) \wedge p) \rightarrow q$ істинна в усіх чотирьох ітераціях, тобто є тавтологією.

Рівняння формули $(p \rightarrow q) \wedge (p \wedge \bar{q})$ має два атоми: p та q . В результаті ми маємо 4 ітерації. Будуємо таблицю істинності.

Таблиця 3.5 – Таблиця значення істинності формули $(p \rightarrow q) \wedge (p \wedge \bar{q})$

p	q	\bar{q}	$p \rightarrow q$	$p \wedge \bar{q}$	$(p \rightarrow q) \wedge (p \wedge \bar{q})$
T	T	F	T	F	F
T	F	T	F	T	F
F	T	F	T	F	F
F	F	T	T	F	F

Отже формула $(p \rightarrow q) \wedge (p \wedge \bar{q})$ хибне в усіх чотирьох ітераціях, тобто є суперечністю.

Завдання 4.

За допомогою таблиць істинності довести твердження: $p \rightarrow q = \bar{p} \vee q$,

$p \rightarrow q \neq q \rightarrow p$.

Розв'язання.

Виходячи з твердження $p \rightarrow q = \bar{p} \vee q$ маємо формулу $(p \rightarrow q) \leftrightarrow (\bar{p} \vee q)$.

Будуємо таблицю істинності для цієї формули.

Таблиця 3.6 – Таблиця істинності формули $(p \rightarrow q) \leftrightarrow (\bar{p} \vee q)$

p	q	\bar{p}	$p \rightarrow q$	$\bar{p} \vee q$	$(p \rightarrow q) \leftrightarrow (\bar{p} \vee q)$
T	T	F	T	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

Отже формула $(p \rightarrow q) \leftrightarrow \bar{p} \vee q$ є тавтологією і це означає формула $p \rightarrow q$ рівносильна $\bar{p} \vee q$.

Аналогічно з твердження $p \rightarrow q \neq q \rightarrow p$ маємо формулу $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$ та будуємо таблицю істинності.

Таблиця 3.7 – Таблиця істинності формули $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$

p	q	$p \rightarrow q$	$q \rightarrow p$	$(p \rightarrow q) \leftrightarrow (q \rightarrow p)$
T	T	T	T	T
T	F	F	T	F
F	T	T	F	F
F	F	T	T	T

Отже формула $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$ істинна тільки 2 із 4 ітерацій і це означає що формула $p \rightarrow q$ не рівносильна формулі $q \rightarrow p$.

3.2 Алгоритм роботи тренажера

Після запуску тренажера, перед користувачем відкривається вікно, в якому можна обрати мову, на якій буде відображено інтерфейс тренажеру. Після вибору мови перед користувачем відкривається головне вікно тренажера, в якому можна передивитися теоретичну частину теми, пройти тестування, та розв'язати приклади з теми. Розглянемо детальніше алгоритм роботи тренажера з прикладами.

Завдання 1.

Крок 1. Користувачу відображається перше запитання «Які з наведених виразів є висловленням?». Нижче відображаються 13 варіантів відповідей:

- 1) Кожне дійсне число задовольняє нерівність $x^2 \geq 0$.
- 2) Число 168 кратне 9.
- 3) Хай живе математика!
- 4) 15 кратне 3, але не кратне 4.
- 5) Чи існує дійсне число, більше за 3 і менше від $\log_2 9$?

- 6) Ця задача легка.
- 7) Існує найбільше просте число.
- 8) Чи правильна велика теорема Ферма?
- 9) Рівняння $x^2 + 7x + 1 = 0$ має хоч один дійсний корінь.
- 10) Розв'язати рівняння $x^2 + 7x + 1 = 0$.
- 11) Розкрийте підручник на сторінці 23.
- 12) Вчитель сказав: «Розкрийте підручник на сторінці 23».
- 13) Якщо $3 < 2$, то $3^2 < 2^2$.

Користувачу потрібно серед запропонованих варіантів відповідей обрати 10 правильних відповідей. Якщо користувач вибере не всі правильні відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти відповіді!», а якщо позначить хоча б один неправильний варіант з'явиться повідомлення: «Обрано хибну відповідь!». Якщо користувач позначив усі правильні варіанти відповіді (а саме: 1, 2, 4, 5, 6, 7, 8, 9, 12, 13) і жодної хибної, то він переходить до кроку 2.

Крок 2. Користувачу відображається друге запитання: «Які з наведених висловлень є істинними?». Нижче відображені варіанти відповіді такі ж, як і на кроці 1.

Користувачу потрібно серед варіантів відповіді виділити тільки ті, в яких зображено істинні висловлення (а саме: 1, 4, 5, 8, 9, 13). Якщо користувач позначить не всі правильні відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти відповіді!», а якщо позначить хоча б один неправильний варіант з'явиться повідомлення: «Обрано хибну відповідь!». Якщо користувач позначив усі правильні відповіді і не допустив жодної помилки, то здійснюється перехід до кроку 3.

Крок 3. Перед користувачем з'являється запитання: «Які з наведених висловлень є хибними?». Нижче відображені варіанти відповіді такі ж, як і на кроці 1.

Користувачу потрібно обрати тільки два варіанти відповіді, в яких зображено хибні висловлення (а саме: 2, 7). Якщо користувач позначить тільки один варіант відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти

відповіді!», а якщо позначити хоч один неправильний варіант, то з'явиться повідомлення «Обрано хибну відповідь!». Коли користувач позначить дві правильні відповіді і жодної неправильної, то він перейде до кроку 4.

Крок 4. Перед користувачем з'являється повідомлення: «Які з наведених висловлень можливі як істинні, так і хибні?». Нижче відображені варіанти відповіді такі ж, як і на кроці 1.

Користувачу потрібно обрати тільки два варіанти відповіді, в яких зображено висловлення які можуть бути як істинні, так і хибні. Якщо користувач позначить тільки один варіант відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти відповіді!», а якщо позначити хоч один неправильний варіант, то з'явиться повідомлення «Обрано хибну відповідь!». Коли користувач позначить тільки 6 та 12 відповідь, то він перейде до кроку 5.

Крок 5. Перед користувачем з'являється повідомлення: «Які з наведених виразів є простим висловленням?». Нижче відображені варіанти відповіді такі ж, як і на кроці 1.

Користувачу потрібно обрати сім варіантів відповіді, в яких зображено прості висловлення. Якщо користувач позначить тільки один варіант відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти відповіді!», а якщо позначити хоч один неправильний варіант, то з'явиться повідомлення «Обрано хибну відповідь!». Коли користувач позначить тільки правильні відповіді (а саме 1, 2, 6, 7, 8, 9, 12), і жодної хибної, то він перейде до кроку 6.

Крок 6. Перед користувачем з'являється повідомлення: «Які з наведених виразів є складним висловленням?». Нижче відображені варіанти відповіді такі ж, як і на кроці 1.

Користувачу потрібно обрати тільки три варіанти відповіді, в яких зображено складні висловлення. Якщо користувач позначить тільки один варіант відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти відповіді!», а якщо позначити хоч один неправильний варіант, то з'явиться повідомлення «Обрано хибну відповідь!». Коли користувач позначить тільки 4, 5 та 13 відповідь, то він перейде до кроку 7.

Крок 7. Перед користувачем з'являється повідомлення: «Які з наведених виразів не є висловленням?». Нижче відображені варіанти відповіді такі ж, як і на кроці 1.

Користувачу потрібно обрати тільки три варіанти відповіді, в яких зображено вирази які не є висловлюваннями. Якщо користувач позначить тільки один варіант відповіді, з'явиться повідомлення «Оберіть усі правильні варіанти відповіді!», а якщо позначити хоч один неправильний варіант, то з'явиться повідомлення «Обрано хибну відповідь!». Коли користувач позначить тільки 3, 10 та 11 відповідь, то він завершить завдання, та перейде до вікна з якого можна почати наступне завдання, або повернутись до головного меню.

Завдання 2.

Крок 1. Користувачу відображається запитання: «Які заперечення висловлювання «Сьогодні середа»?» Нижче зображені варіанти відповіді:

- 1) це так що, сьогодні середа;
- 2) настала середа;
- 3) це не так що, сьогодні середа;
- 4) сьогодні робочий день.

Користувачу потрібно обрати одну правильну відповідь. Якщо користувач обирає третій варіант відповіді, то він переходить до кроку 2. Якщо обрано іншу відповідь, то з'явиться повідомлення «Не вірно! Це не заперечення!» і користувач залишиться на тому ж самому вікні.

Крок 2. Користувачу відображається запитання: «В яких випадках речення, що пов'язані з часовою змінною, це висловлювання?» Нижче зображені варіанти відповіді:

- 1) речення в якому визначено відрізок часу;
- 2) речення в якому визначено момент часу;
- 3) речення в якому визначено діапазон часу;
- 4) речення в якому визначено приблизний час.

Користувачу потрібно обрати одну правильну відповідь. Якщо користувач обирає другий варіант відповіді, то він переходить до кроку 3. Якщо обрано іншу

відповідь, то з'явиться повідомлення «Відповідь не вірна!» і користувач залишиться на тому ж самому вікні.

Крок 3. Користувач відображається запитання: «Яких речень, раніше згадане правило теж стосується?» Нижче зображені варіанти відповіді:

- 1) речень, які характеризують місце чи особу;
- 2) речень, які описують приблизне місце;
- 3) речень, які характеризують стан невідомого предмета;
- 4) речень які задають питання;

Користувачу потрібно обрати одну правильну відповідь. Якщо користувач обирає перший варіант відповіді, то він переходить до кроку 4. Якщо обрано іншу відповідь, то з'явиться повідомлення «Відповідь не вірна!» і користувач залишиться на тому ж самому вікні.

Крок 4. Користувачу відображається запитання: «Знайти кон'юнкцію висловлень «Сьогодні п'ятниця» і «Сьогодні падає дощ»?» Нижче зображені варіанти відповіді:

- 1) сьогодні п'ятниця і падає дощ;
- 2) сьогодні п'ятниця і сьогодні не падає дощ;
- 3) сьогодні не п'ятниця і сьогодні падає дощ;
- 4) сьогодні не п'ятниця і сьогодні не падає дощ.

Користувачу потрібно обрати одну правильну відповідь. Якщо користувач обирає другий варіант відповіді, то він переходить до кроку 5. Якщо обрано іншу відповідь, то з'явиться повідомлення «Відповідь не вірна!» і користувач залишиться на тому ж самому вікні.

Крок 5. Користувач відображається запитання: «В якому випадку висловлення «Сьогодні п'ятниця і сьогодні падає дощ» істинне?» Нижче зображені варіанти відповіді:

- 1) в дощовий день;
- 2) в недощову п'ятниця;
- 3) в п'ятницю;
- 4) в дощову п'ятницю.

Користувачу потрібно обрати одну правильну відповідь. Якщо користувач обирає четвертий варіант відповіді, то користувач виконає завдання і з'явиться вікно за допомогою якого можна перейти до наступного завдання, або в головне меню. Якщо обрано іншу відповідь, то з'явиться повідомлення «Відповідь не вірна!» і користувач залишиться на тому ж самому вікні.

Завдання 3.

Крок 1. Перед користувачем з'являється завдання «Знайти значення істинності формул алгебри висловлень $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$, $((p \rightarrow q) \wedge p) \rightarrow q$, $(p \rightarrow q) \wedge (p \wedge \bar{q})$ у всіх інтерпретаціях» і кнопка «Розпочати». Після натискання на кнопку користувач переходить до кроку 2.

Крок 2. Перед користувачем з'являється повідомлення: «Позаяк кожному з атомів p , q й r формули $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$ можна надати два значення (F та T). Скільки ітерацій має дана формула?». Під повідомленням є поле з написом «Кількість ітерацій:» і користувачу потрібно ввести кількість можливих ітерацій. Якщо користувач поставить 8 ітерацій, то він перейде до кроку 3. Якщо користувач введе будь-яке інше число, то з'явиться повідомлення: «Не правильно! В нас є три атоми, кожне з яких може мати по два значення!»

Крок 3. Перед користувачем з'являється повідомлення: «Маємо таблицю. В таблиці виставіть значення колонки \bar{r} », та з'явиться таблиця, де значення атомів p , q та r вже виставлено та потрібно вручну виставити значення колонки \bar{r} . Якщо користувач правильно позначив значення комірок таблиці, то він потрапляє до кроку 4, а якщо ні то з'явиться повідомлення: «В якомусь полі допущена помилка! Згадайте про правило заперечення!»

Крок 4. Перед користувачем з'являється повідомлення: «Знайдемо ліву частину виразу $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$. В таблиці виставіть значення колонки $(p \wedge q)$ », та з'явиться таблиця, де значення попередніх комірок вже виставлено та потрібно вручну виставити значення колонки $(p \wedge q)$. Якщо користувач правильно позначив значення комірок таблиці, то він потрапляє до кроку 5, а

якщо ні то з'явиться повідомлення: «В якомусь полі допущена помилка! Кон'юнкція істинна тільки тоді, коли істинні обидва висловлювання!»

Крок 5. Перед користувачем з'являється повідомлення: «Знайдемо праву половину виразу $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$. В таблиці виставіть значення колонки $(p \leftrightarrow q)$ », та з'явиться таблиця, де значення попередніх комірок вже виставлено та потрібно вручну виставити значення колонки $(p \leftrightarrow q)$. Якщо користувач правильно позначив значення комірок таблиці, то він потрапляє до кроку 6, а якщо ні то з'явиться повідомлення: «В якомусь полі допущена помилка! Рівносильні висловлення тільки тоді, коли обидва висловлення або одночасно істинні або одночасно хибні!»

Крок 6. Перед користувачем з'являється повідомлення: «Знайдемо значення ітерацій формули $(p \wedge q) \rightarrow (\bar{r} p \leftrightarrow)$ », та з'явиться таблиця, де значення попередніх комірок вже виставлено та потрібно вручну виставити значення колонки $\bar{r} (p \wedge q) \rightarrow (p \leftrightarrow)$. Якщо користувач правильно позначив значення комірок таблиці, то він потрапляє до кроку 7, а якщо ні то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення імплікації хибне тоді і тільки тоді, коли перше значення істинне, а друге хибне!»

Крок 7. Перед користувачем з'являється повідомлення: «Формула $((p \rightarrow q) \wedge p) \rightarrow q$ має два атоми q та p. Скільки ітерацій матиме ця формула?». Під повідомленням є поле з написом «Кількість ітерацій:» і користувачу потрібно ввести кількість можливих ітерацій. Якщо користувач поставить 4 ітерацій, то він перейде до кроку 8. Якщо користувач введе будь-яке інше число, то з'явиться повідомлення: «Не правильно! В нас є два атоми, кожне з яких може мати по два значення!».

Крок 8. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $((p \rightarrow q) \wedge p) \rightarrow q$. Для початку знайдіть значення $(p \rightarrow q)$ ». Нижче зображена таблиця в якій колонки p та q вже заповнені, а колонку $(p \rightarrow q)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 9, а якщо ні, то з'явиться повідомлення: «В якомусь полі

допущена помилка! Значення імплікації хибне тоді і тільки тоді, коли перше значення істинне, а друге хибне!».

Крок 9. Перед користувачем з'являється повідомлення: «Знайдіть значення: $((p \rightarrow q) \wedge p)$ ». Нижче зображена таблиця в якій попередні колонки вже заповнені, а колонку $((p \rightarrow q) \wedge p)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 10, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Кон'юнкція істинна тільки тоді, коли істинні обидва висловлювання!».

Крок 10. Перед користувачем з'являється повідомлення: «Знайдемо значення істинності формули $((p \rightarrow q) \wedge p) \rightarrow q$ ». Нижче зображена таблиця в якій попередні колонки вже заповнені, а колонку $((p \rightarrow q) \wedge p) \rightarrow q$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 11, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення імплікації хибне тоді і тільки тоді, коли перше значення істинне, а друге хибне!».

Крок 11. Перед користувачем з'являється повідомлення з питанням: «Формула $((p \rightarrow q) \wedge p) \rightarrow q$ істинна в усіх чотирьох ітераціях. Як її можна назвати?», а нижче варіанти відповіді:

- 1) істина;
- 2) компромісна;
- 3) тавтологія;
- 4) суперечність.

Якщо користувач обирає третій варіант відповіді, то він переходить до кроку 12. Якщо було обрано будь-яку іншу відповідь, то з'являється повідомлення: «Відповідь не вірна!».

Крок 12. Перед користувачем з'являється повідомлення: «Формула $(p \rightarrow q) \wedge (p \wedge \bar{q})$ має два атоми q та p . Скільки ітерацій матиме ця формула?». Під повідомленням є поле з написом «Кількість ітерацій:» і користувачу потрібно ввести кількість можливих ітерацій. Якщо користувач поставить 4 ітерацій, то він перейде до кроку 13. Якщо користувач введе будь-яке інше число, то з'явиться

повідомлення: «Не правильно! В нас є два атоми, кожне з яких може мати по два значення!».

Крок 13. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $(p \rightarrow q) \wedge (p \wedge \bar{q})$. Для початку знайдіть значення \bar{q} ». Нижче зображена таблиця в якій колонки p та q вже заповнені, а колонку \bar{q} потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 14, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Згадайте правило заперечення!».

Крок 14. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $(p \rightarrow q)$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $(p \rightarrow q)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 15, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення імплікації хибне тільки тоді, коли перше значення істинне, а друге хибне!».

Крок 15. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $(p \wedge \bar{q})$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $(p \wedge \bar{q})$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 16, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Кон'юнкція істинна тоді й тільки тоді!».

Крок 16. Перед користувачами з'являється повідомлення: « $(p \rightarrow q) \wedge (\bar{q} \wedge \quad)$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $(p \rightarrow \bar{q}) \wedge (p \wedge \quad)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 17, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення імплікації хибне тільки тоді, коли перше значення істинне, а друге хибне!».

Крок 17. Перед користувачем з'являється повідомлення з питанням: «Формула $(p \rightarrow q) \wedge \bar{q} \wedge \quad$ хибна в усіх чотирьох ітераціях. Як її можна назвати?», а нижче варіанти відповіді:

- 1) суперечність

- 2) істина
- 3) компромісна
- 4) тавтологія

Якщо користувач вибере перший варіант відповіді, то він завершить завдання та перейде до вікна, через яке можна почати наступне завдання, або перейти до головного меню.

Завдання 4.

Крок 1. Перед користувачем з'являється повідомлення : «За допомогою таблиць істинності довести твердження для формул: $p \rightarrow q : \bar{p} \vee q$, $p \rightarrow q \neq q \rightarrow p$ » та кнопка «Розпочати». Після натискання на кнопку користувач переходить до кроку 2.

Крок 2. Перед користувачем з'являється повідомлення: «Почнемо з твердження: $p \rightarrow q : \bar{p} \vee q$. Маємо формулу: $(p \rightarrow q) \bar{p} \rightarrow (\bar{p} \vee q)$ яка має два атоми q та p . Скільки ітерацій має дана формула?». Під повідомленням є поле з написом «Кількість ітерацій:» і користувачу потрібно ввести кількість можливих ітерацій. Якщо користувач поставить 4 ітерацій, то він перейде до кроку 3. Якщо користувач введе будь-яке інше число, то з'явиться повідомлення: «Не правильно! В нас є два атоми, кожне з яких може мати по два значення!»

Крок 3. Перед користувачем з'являється повідомлення: «Для початку знайдіть значення \bar{p} ». Нижче зображена таблиця в якій колонки p та q вже заповнені, а колонка \bar{p} потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 4, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Згадайте правило заперечення!».

Крок 4. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $(p \rightarrow q)$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $(p \rightarrow q)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 5, а якщо ні, то з'явиться

повідомлення: «В якомусь полі допущена помилка! Значення імплікації хибне тільки тоді, коли перше значення істинне, а друге хибне!».

Крок 5. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $(\bar{p} \vee q)$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $(\bar{p} \vee q)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 6, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення диз'юнкції хибне тільки тоді, коли перше і друге значення хибні!».

Крок 6. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу: $(p \rightarrow q) \leftrightarrow (\bar{p} \vee q)$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $(p \rightarrow q) \leftrightarrow (\bar{p} \vee q)$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 7, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Рівносильні висловлення тільки тоді, коли обидва висловлення або одночасно істинні або одночасно хибні!».

Крок 7. Перед користувачем з'являється повідомлення: «Отже якщо формула $(p \rightarrow q) \leftrightarrow (\bar{p} \vee q)$ істинна в усіх чотирьох ітераціях, то рівняння $\bar{p} \rightarrow q = \bar{p} \vee q$ є правильним?» та варіанти відповіді:

- 1) так;
- 2) ні.

Якщо користувач вибере перший варіант відповіді, переходить до кроку 8, а якщо ні, то з'явиться повідомлення: «Відповідь не вірна! Адже якщо рівняння істинне в усіх ітераціях, то рівняння правильне!»

Крок 8. Перед користувачем з'являється повідомлення: «Почнемо з твердження: $p \rightarrow q \neq q \rightarrow p$. Маємо формулу: $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$ яка має два атоми q та p . Скільки ітерацій має дана формула?». Під повідомленням є поле з написом «Кількість ітерацій:» і користувачу потрібно ввести кількість можливих ітерацій. Якщо користувач поставить 4 ітерацій, то він перейде до кроку 9. Якщо користувач введе будь-яке інше число, то з'явиться повідомлення: «Не правильно! В нас є два атоми, кожне з яких може мати по два значення!»

Крок 9. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу $p \rightarrow q$ ». Нижче зображена таблиця в якій колонки p та q вже заповнені, а колонку $p \rightarrow q$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 10, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення диз'юнкції хибне тільки тоді, коли перше і друге значення хибні!».

Крок 10. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу $q \rightarrow p$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку $q \rightarrow p$ потрібно заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 11, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Значення диз'юнкції хибне тільки тоді, коли перше і друге значення хибні!».

Крок 11. Перед користувачем з'являється повідомлення: «Тепер знайдемо значення виразу $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$ ». Нижче зображена таблиця в попередні колонки вже заповнені, а колонку потрібно $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$ заповнити вручну. Якщо користувач все заповнив правильно, то він переходить до кроку 12, а якщо ні, то з'явиться повідомлення: «В якомусь полі допущена помилка! Рівносильні висловлення тільки тоді, коли обидва висловлення або одночасно істинні або одночасно хибні!».

Крок 12. Перед користувачем з'являється повідомлення: «Отже якщо формула $p \rightarrow q \neq q \rightarrow p$ істинна в усіх чотирьох ітераціях, то рівняння $(p \rightarrow q) \leftrightarrow (q \rightarrow p)$ є правильним?» та варіанти відповіді:

3) так;

4) ні.

Якщо користувач вибере другий варіант відповіді, переходить до кроку 8, а якщо ні, то він завершить завдання та перейде до вікна, через яке можна перейти до головного меню.

3.3 Розробка блок-схеми алгоритму роботи навчального тренажера

Щоб краще зрозуміти алгоритм роботи тренажеру з теми «Алгебра висловлень» потрібно продемонструвати його у вигляді блок-схеми. Так як тренажер складається з кількох розділів, а ті в свою чергу з кількох підрозділів. Це допоможе більш детально показати принцип роботи тренажера, та полегшити його програмну реалізацію.

Розберемо роботу тренажера. Спочатку перед користувачем з'являється головне вікно програми, в якому ми можемо обрати три варіанти дій, а саме:

- переглянути лекційний матеріал;
- пройти тестові завдання по лекційному матеріалу;
- вирішити декілька практичних завдань з теми.

Якщо обрати лекційний матеріал, то перед користувачем з'являється вікно яке відображається вікно в якому знаходиться теоретичний матеріал з теми «Алгебра висловлень».

Якщо обрати тестові завдання, то можна вибрати режим проходження завдань, а саме: з підказками або на оцінку. Різниця між режимами полягає в тому, що в разі неправильної відповіді на поставлене запитання відображається вікно, у якому повідомляється що відповідь неправильна і підказка, яка допоможе обрати правильну відповідь. У другому режимі, якщо користувач помилиться, то він все одно перейде до наступного питання, але бал за відповідь не нараховується, а в кінці буде відображено підсумковий бал.

Якщо користувач вибере практичні завдання, то перед ним з'явиться список із чотирьох завдань. Кожне завдання вирішується в покроковому режимі. Якщо допустити помилку, то з'явиться повідомлення, що було допущено помилку і з'явиться підказка, яка допоможе вирішити етап завдання, на якому виникли складнощі. Після вирішення завдання, з'являється пропозиція перейти до наступного завдання або повернутись у головне меню.

Блок-схема алгоритму роботи навчального тренажера з теми «Алгебра висловлень» зображена на рисунку 3.1.

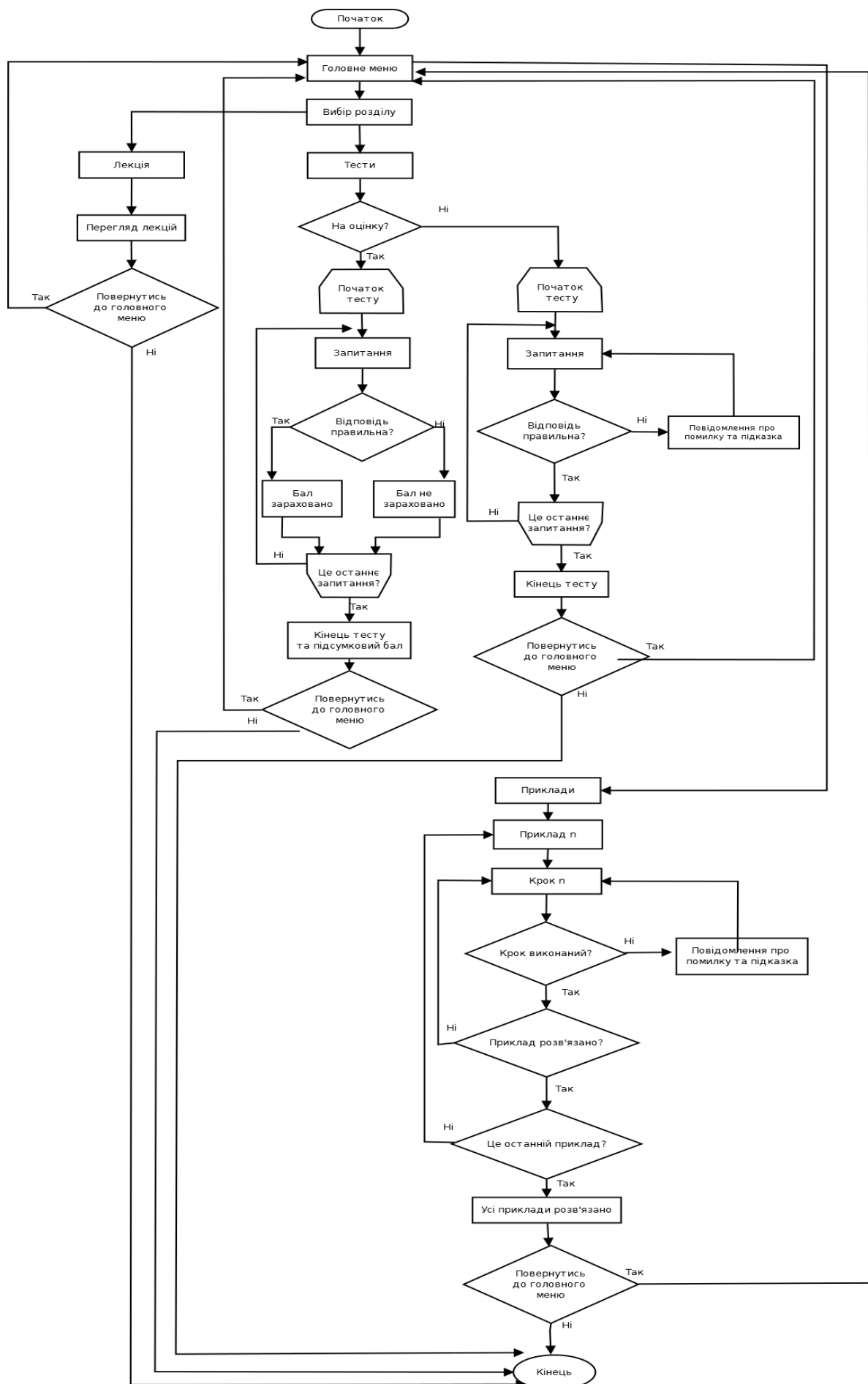


Рис. 3.1 – Блок-схема роботи алгоритму тренажера

3.3 Обґрунтування вибору програмних засобів

Обрана мова програмування для створення навчального тренажера повинна відповідати наступним вимогам:

- наявність можливості конструювання графічного інтерфейсу для полегшення користувачу роботи з тренажером;
- можливість інтегрувати написаний додаток в систему Moodle;
- сумісність з сучасними операційними системами;
- можливість найбільш точно реалізувати алгоритм роботи тренажера;
- зручність написання додатків на обраній мові.

Алгоритм роботи тренажера вирішено реалізувати у вигляді комп'ютерної програми. Для написання програми було обрано мову Java SE 11.

Java – це сильно типізована об'єктно-орієнтована мова програмування. Працює на різних платформах, таких як Windows, Mac OS, а також різноманітних версіях Unix.

Для створення графічного інтерфейсу програми було обрано технологію JavaFX 2.0. JavaFX – це платформа для створення інтерфейсу, яка дозволяє будувати уніфіковані додатки з насиченим графічним інтерфейсом користувача для безпосереднього запуску з-під операційних систем, роботи в браузері та на мобільних пристроях. Вона була покликана замінити собою бібліотеку Swing. Платформа JavaFX конкурує з Microsoft Silverlight, Adobe Flash та аналогічними системами.

Платформа JavaFX це нова ступінь в розробці користувацького інтерфейсу який дає розробнику можливість простої розробки RIA («багатих інтернет-додатків»), які працюють на різних платформах. Заснована на Java, платформа забезпечує широку графічну та медіа підтримку, яка заснована на високопродуктивному графічному та медіа рушії, який значно спрощує розробку клієнтських додатків.

3.4 Висновки за розділом 3

У результаті виконання теоретичної частини роботи можна зробити наступні висновки:

1. Щоб програмно реалізувати навчальний тренажер, було розглянуто теоретичні відомості з теми «Алгебра висловлювань», на основі яких було створено теоретичний матеріал. Розглянуто розв'язок задач та на їх основі створено покроковий алгоритм розв'язання.

2. За допомогою алгоритму розв'язку задач, було розроблено блок-схему алгоритму тренажера.

3. Сформовані основні вимоги, яким потрібно відповідати середовище розробки, яку обрано для програмної реалізації для навчального тренажера, а саме: простота, зрозумілість, можливість написання графічного інтерфейсу, можливість інтеграції в Moodle. Перерахованим вище вимогам відповідає мова Java, яку було обрано для реалізації навчального тренажера з теми «Алгебра висловлювань».

4 ПРАКТИЧНА ЧАСТИНА

4.1 Опис процесу програмної реалізації

У даному підрозділі розглянуто основний принцип програмної реалізації тренажера з теми «Алгебра висловлень».

Програма є віконним додатком з графічним інтерфейсом, який складається з вікон. Враховуючи той факт, що розв'язання задачі складається з декількох кроків, які не завжди схожі між собою, то кожен крок має своє вікно. Вікна програми були спроектовані за допомогою графічного інтерфейсу JavaFX. Інтерфейс програми описаний на мові розмітки XML на який посилається файл java. Завдяки такому методу дуже інтерфейс вікна. Також за допомогою додатку Scene Builder можна переглянути майбутній інтерфейс, ще до компіляції програми (рис. 4.1).

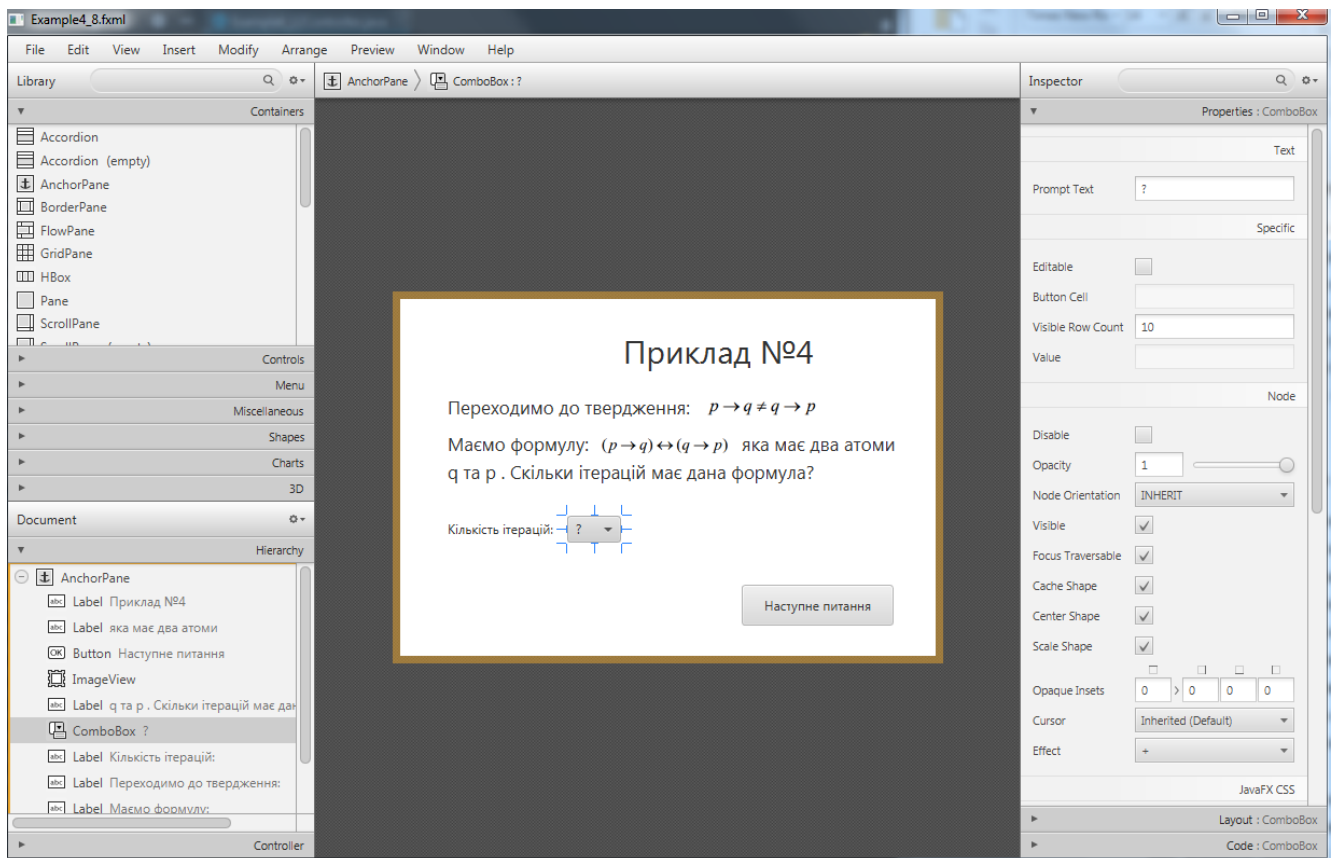


Рисунок 4.1 – Розробка інтерфейсу в програмі Scene Builder

Основний код написаний в файлах типу *файл_програми.java*. У таких файлах міститься код, за допомогою якого файл з розширенням *java* посилається на файл який містить інтерфейс з розширенням *fxml*.

Приклад такого посилання має вигляд:

```
//Підключаємо бібліотеки
Import javafx.fxml.FXML;
Import javafx.fxml.FXMLLoader
//Елементи інтерфейсу які будуть запрограмовані в коді програми
@FXML
private Label Label1
@FXML
private Button Button1
@FXML
private Button Button2
Void initialize() {
    Assert Label1 !=null : "fx:id=\"bt\" was not injected: check your FXML file
'Program_Interface.fxml'.";
    Assert Button1 !=null : "fx:id=\"bt\" was not injected: check your FXML file
'Program_Interface.fxml'.";
    Assert Button2 !=null : "fx:id=\"bt\" was not injected: check your FXML file
'Program_Interface.fxml'.";
```

Тепер описані елементи інтерфейсу в фалі з розширенням *fxml* можна кодувати у файлі з розширенням *java*.

Основне керування здійснюється за допомогою кнопок які здійснюють перехід в інші вікна за допомогою наступного коду:

```
FXMLLoader loader = new FXMLLoader();
loader.setLocation(getClass().getResource("Example4Final.fxml"));
```

В основному тести базуються на таких системах керування CheckBox, RadioButton та ComboBox в яких користувач обирає правильний варіант відповіді та натискає на кнопку, після чого перевіряється чи правильно було дано відповідь на запитання. Якщо відповідь дано було не правильно, то з'являється модальне

вікно з повідомленням про помилку та підказкою яка допоможе обрати правильну відповідь.

4.2 Опис програми

Розглянемо програмний код на основі одного з вікон завдання №3, а саме його сьомого кроку, де зображено таблицю істинності розв’язуваного рівняння і де користувачу потрібно заповнити цю таблицю.

Спочатку було створено вікно інтерфейсу для інтерфейсу за допомогою додатку Scene Builder (рис. 4.2).

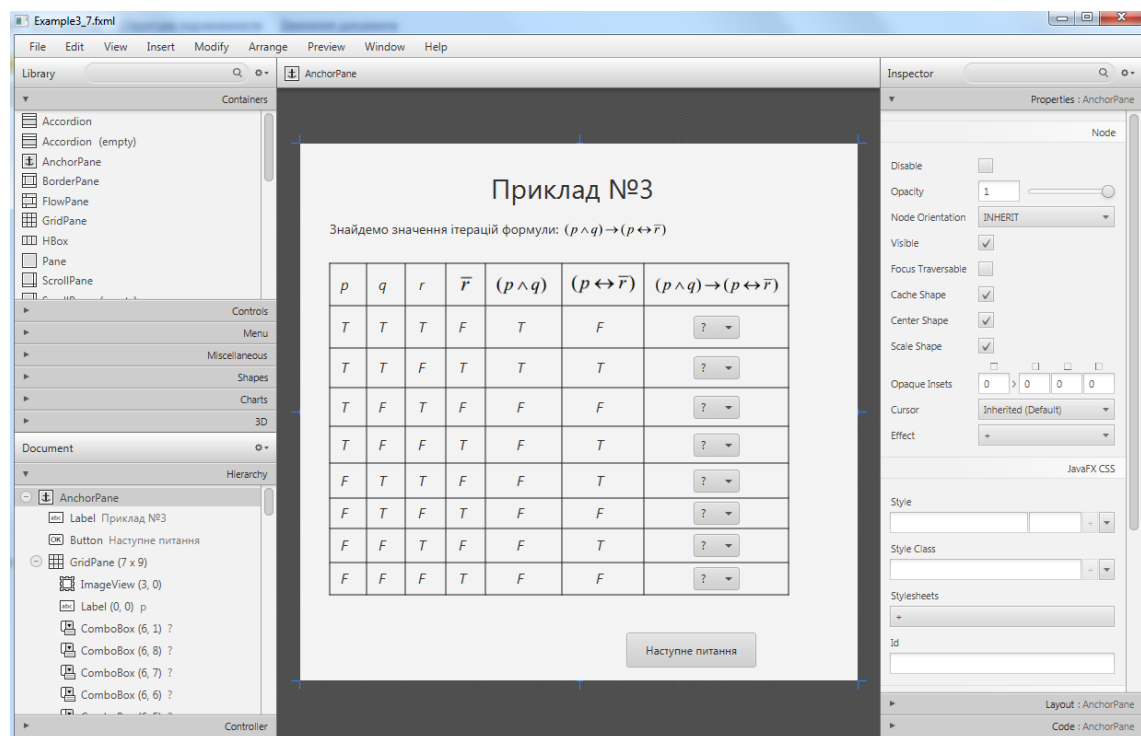


Рисунок 4.2 – Вікно програми з завданням №3

На цьому етапі також розмічаємо такі елементи як label, button та image. Таблиця створена завдяки елементу GridPane. Розмітивши та налаштувавши всі елементи, відкриваємо основний код цієї форми.

Перш за все слід імпортувати бібліотеки доданих елементів, в нашому випадку це елементи *control*, *layout*, *text*, *image*. В результаті маємо наступний код:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<?import javafx.scene.image.*?>
<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

```

Наступним кроком йде розмітка робочого вікна яке має назву *AnchorPane*. Всередині нього розмічаємо основні властивості вікна, а саме висоту, ширину, та java файл з яким буде взаємодіяти інтерфейс. Таким чином елемент *AnchorPane* розмічений в коді так:

```

<AnchorPane prefHeight="583.0" prefWidth="605.0" xmlns="http://javafx.com/javafx/8"
xmlns:fx="http://javafx.com/fxml/1" fx:controller="sample.Example3_7Controller">
    <children>
        //Тут розміщуються основні елементи інтерфейсу
    </children>
</AnchorPane>

```

Враховуючи що *AnchorPane* є батьківським елементом, то усі інші керуючі та візуальні елементи інтерфейсу будуть розмічені в класі *children*.

Всередині розмічаємо елемент *label* та вказуємо його властивості, а саме: положення, розмір, та наповнення. Маємо такий код:

```

<Label layoutX="32.0" layoutY="83.0" prefHeight="20.0" prefWidth="264.0"
text="Знайдемо значення ітерацій формули:">
    <font>
        <Font size="14.0" />
    </font>
</Label>

```

Наступним кроком потрібно вказати властивості таблиці, яка розміщена під текстом. Елемент який відповідає за таблицю називається *GridPane*. В середині якого розміщені такі елементи як *ColumnConstraints* яка відповідають за розмітку колонок таблиці та *RowConstraints*, які відповідають за рядки таблиці. В середині кожного з вище перерахованих елементів потрібно вказати властивості. Розмічена таблиця разом з розміченими елементами має такий вигляд:

```

<GridPane alignment="CENTER" focusTraversable="true" gridLinesVisible="true"
layoutX="32.0" layoutY="130.0" prefHeight="360.0" prefWidth="498.0">
    <columnConstraints>

```

```

        <ColumnConstraints hgrow="SOMETIMES" maxWidth="192.0" minWidth="0.0"
prefWidth="61.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="495.0" minWidth="10.0"
prefWidth="63.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="603.0" minWidth="10.0"
prefWidth="64.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="599.0" minWidth="10.0"
prefWidth="63.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="599.0" minWidth="10.0"
prefWidth="103.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="599.0" minWidth="10.0"
prefWidth="109.0" />
        <ColumnConstraints hgrow="SOMETIMES" maxWidth="599.0" minWidth="10.0"
prefWidth="177.0" />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints maxHeight="52.0" minHeight="10.0" prefHeight="42.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="53.0" minHeight="10.0" prefHeight="42.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="52.0" minHeight="10.0" prefHeight="39.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="53.0" minHeight="10.0" prefHeight="38.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="52.0" minHeight="10.0" prefHeight="36.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="90.0" minHeight="10.0" prefHeight="34.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="67.0" minHeight="10.0" prefHeight="30.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="67.0" minHeight="10.0" prefHeight="34.0"
vgrow="SOMETIMES" />
        <RowConstraints maxHeight="67.0" minHeight="10.0" prefHeight="32.0"
vgrow="SOMETIMES" />
    </rowConstraints>

```

```

<children>
||Тут розміщені елементи які розміщені в середині таблиці
</children>
</GridPane>

```

Елементи які розміщуються в середині таблиці *GridPain* потрібно розміщувати в середині поля *children* вказуючи такі властивості як *GridPain.columnIndex="N"* та *GridPane.rowIndex="N"*, які відповідно означають комірку колонки та рядка, а на місті *N* повинно бути число. Розглянемо на прикладі розміщення елементу *label* в середині таблиці:

```

<Label    prefHeight="25.0"    prefWidth="16.0"    text="F"    textAlignment="CENTER"
GridPane.columnIndex="4" GridPane.halignment="CENTER" GridPane.rowIndex="8">
    <font>
        <Font name="System Italic" size="16.0" />
    </font>
</Label>

```

З вище описаного коду видно що елемент *Label* має властивість *GridPane.columnIndex="4"* та *GridPane.rowIndex="8"*, це означає що елемент знаходиться в комірці яку перетинає четверта колонка та восьмий рядок. Властивість *GridPane.halignment="CENTER"* означає що елементи вирівняно по центру комірки.

Елементи керування мають особливу властивість як *fx:id=""*, де в яке містить унікальне ідентифікуюче ім'я, за допомогою якого елемент може отримувати код з *java* файлу. Наприклад кожен *ComboBox*, який відображений в таблиці, має своє власне ім'я, як зображено в коді:

```

<ComboBox    fx:id="ComboBox1"    promptText="?"    GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="1" />
<ComboBox    fx:id="ComboBox8"    promptText="?"    GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="8" />
<ComboBox    fx:id="ComboBox7"    promptText="?"    GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="7" />
<ComboBox    fx:id="ComboBox6"    promptText="?"    GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="6" />

```

```

        <ComboBox fx:id="ComboBox5" promptText="?" GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="5" />
        <ComboBox fx:id="ComboBox4" promptText="?" GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="4" />
        <ComboBox fx:id="ComboBox3" promptText="?" GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="3" />
        <ComboBox fx:id="ComboBox2" promptText="?" GridPane.columnIndex="6"
GridPane.halignment="CENTER" GridPane.rowIndex="2" />

```

Також своє власне ідентифікаційне ім'я має кнопка, на яку користувач натискає, коли дає відповідь:

```

<Button fx:id="bt" layoutX="354.0" layoutY="531.0" mnemonicParsing="false"
prefHeight="38.0" prefWidth="141.0" text="Наступне питання" />

```

Переходимо до файлу який відповідає за роботу самої програми, а саме файлу java. Файл починається з того що нам потрібно спочатку вказати до якої бібліотеки класів належить файл і повинен бути написаний в першому некоментованому рядку, а потім які класи потрібно імпортувати в файл. Код приймає наступний вигляд:

```

package sample;

import javafx.collections.FXCollections;
import javafx.collections.ObservableList;
import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.geometry.Orientation;
import javafx.geometry.Pos;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.ComboBox;
import javafx.scene.control.Label;
import javafx.scene.layout.FlowPane;
import javafx.stage.Modality;

```

```
import javafx.stage.Stage;
```

```
import java.io.IOException;
```

```
import java.net.URL;
```

```
import java.util.ResourceBundle;
```

Далі описуємо сам клас та об'являємо керуючі елементи з fxml файлу.

Клас об'являємо за допомогою команди *public class* потім вказується назва класу та відкриваємо фігурну дужку. Опісля об'являємо усі елементи що ми розмічали в файлі з розширенням fxml, пишучи *@FXML* та з нового рядка оголошуємо сам елемент *privat тип_елементу назва_елементу*. В програмному коді, це виглядає наступним чином:

```
public class Example3_7Controller<string> {
```

```
    @FXML
```

```
    private ResourceBundle resources;
```

```
    @FXML
```

```
    private URL location;
```

```
    @FXML
```

```
    private Button bt;
```

```
    @FXML
```

```
    private ComboBox<String> ComboBox6;
```

```
    @FXML
```

```
    private ComboBox<String> ComboBox5;
```

```
    @FXML
```

```
    private ComboBox<String> ComboBox8;
```

```
    @FXML
```

```
    private ComboBox<String> ComboBox7;
```

@FXML

private ComboBox<String> ComboBox2;

@FXML

private ComboBox<String> ComboBox1;

@FXML

private ComboBox<String> ComboBox4;

@FXML

private ComboBox<String> ComboBox3;

Далі починається основне тіло файлу, де містяться основний код раніше розміченого вікна, написаний код елементів керування та реалізований алгоритм роботи завдання (а точніше одного з кроків завдання). Спочатку йде ініціалізація роботи коду за допомогою строки «*void initialize() }*». Тепер коли пішов сам код який відповідає за роботу вікна, потрібно позначити керуючі елементи форми та вказати які саме елементи вікна будуть керуватися з нашого файлу за допомогою команди «*assert IDНазва_елементу!= null : "fx:id=\"IDНазва_елементу\" was not injected: check your FXML file 'Файл_з_графічним_інтерфейсом.fxml'."*»». Після розмітки кожного із керуючих елементів описуємо можливі значення елементу *ComboBox*. За допомогою команди *ObservableList* вказуємо що в елементі *ComboBox* може бути тільки два значення Т або F. Рядок з цим кодом має вигляд:

ObservableList<String> value = FXCollections.observableArrayList("T", "F");

Потім прописуємо кожному елементу *ComboBox* значення *value*. Вигляд коду на якому прописані властивості першого *ComboBox*:

ComboBox1.setItems(value); //задаємо значення для списку можливих варіантів в ComboBox

ComboBox1.setValue("?"); //Задаємо значення за замовчуванням

Потім потрібно запрограмувати кнопку, при натисканні на яку відбувався алгоритм перевірки правильності відповіді, та перехід на наступний крок. Для програмування кнопки використано подію «*Ім'я_елементу.setOnAction(event ->*

`{}>>?` де в фігурних дужках записано код який відбувається під час взаємодії з елементом (в нашому випадку натискання кнопки).

Роботу алгоритму реалізовано за допомогою оператора розгалудження *if*. На прикладі *ComboBox*, це алгоритм має такий вигляд:

```
if (ComboBox.getValue().equals("T")){ //Умова оператора розгалуження
//Якщо значення ComboBox дорівнює T, то виконується код який має бути тут
} else {
//але якщо значення не дорівнює T, то виконується це код
}
```

В нашому випадку, якщо користувач відповів правильно то він переходить до наступного кроку. Наступний крок завантажує fxml файл, а той у свою чергу завантажує java файл і таким чином перед користувачем відкривається нове вікно з наступним кроком. А в разі неправильної відповіді перед користувачем з'являється модальне вікно з повідомленням про помилку та підказкою. У програмному коді це виглядає так:

```
if (ComboBox.getValue().equals("T")){
    bt.getScene().getWindow().hide();
    FXMLLoader loader = new FXMLLoader();
    loader.setLocation(getClass().getResource("Example3_7.fxml"));

    try {
        loader.load();
    } catch (IOException e) {
        e.printStackTrace();
    }

    Parent root = loader.getRoot();
    Stage Practic = new Stage();
    Practic.setTitle("Практична з теми Алгебра висловлювань");
    Practic.setScene(new Scene(root));
    Practic.show();
}
}else {
```



```

Label lb = new Label("В якомусь полі допущена помилка\n
Значення імплікації хибне тільки тоді, коли перше значення істинне, а друге хибне! \n");
Button bt = new Button ("Повренутись назад!");
bt.setOnAction(new EventHandler<ActionEvent>() {
    @Override
    public void handle(ActionEvent event) {
        bt.getScene().getWindow().hide();
    }
});
FlowPane root = new FlowPane(Orientation.VERTICAL, lb, bt);
root.setAlignment(Pos.CENTER);
Scene secondScene = new Scene(root, 500, 500);
// New window (Stage)
Stage newWindow = new Stage();
newWindow.setTitle("Second Stage");
newWindow.setScene(secondScene);
// Specifies the modality for new window.
newWindow.initModality(Modality.APPLICATION_MODAL);
newWindow.show();
}

```

За схожим принципом працюють і інші вікна програми, але з тою різницею що вирішення того чи іншого завдання може відрізнятися в залежності від умов. Наприклад завдання де потрібно обрати одну відповідь з декількох реалізовано за допомогою елементів *RadioButton* які в свою чергу об'єднані в групу *ToggleGroup*. За допомогою об'єднання в групу, можна обрати тільки одну відповідь. Візуально такий код виглядає так:

```

ToggleGroup group = new Toggle Group(); //Група яка об'єднує елементи
RadioButton1.setToggleGroup(group);           //Перший елемент
RadioButton2.setToggleGroup(group);           //Другий елемент

bt.setOnAction(event ->{                       // дія яка відбувається під час натискання на
кнопку
    if (RadioButton1.isSelected()){             //Якщо обрано перший елемент
        {
            //то відбувається цей код

```

```

    }
    } else if (RadioButton2.isSelected()){ // Якщо обрано другий елемент
        //то відбувається це код
    } else { //якщо не обрано елемент
        //відбувається це код
    }
});

```

А для завдань де потрібно обрати декілька відповідей із багатьох використовується елемент `CheckBox`, але він зручніше сприймається для позначення задач, де потрібно обрати декілька варіанті відповіді замість одного.

4.3 Перевірка валідності. Дослідження можливостей програмної реалізації.

Для перевірки працездатності програми тестувалися всі вікна програми. При запуску програмного продукту з'явиться стартове вікно, де запропонують обрати мову тренажера (рис. 4.3). В цьому розділі описано тестування української частини тренажера. Тестування англійської версії нічим не відрізняється від україномовної версії.

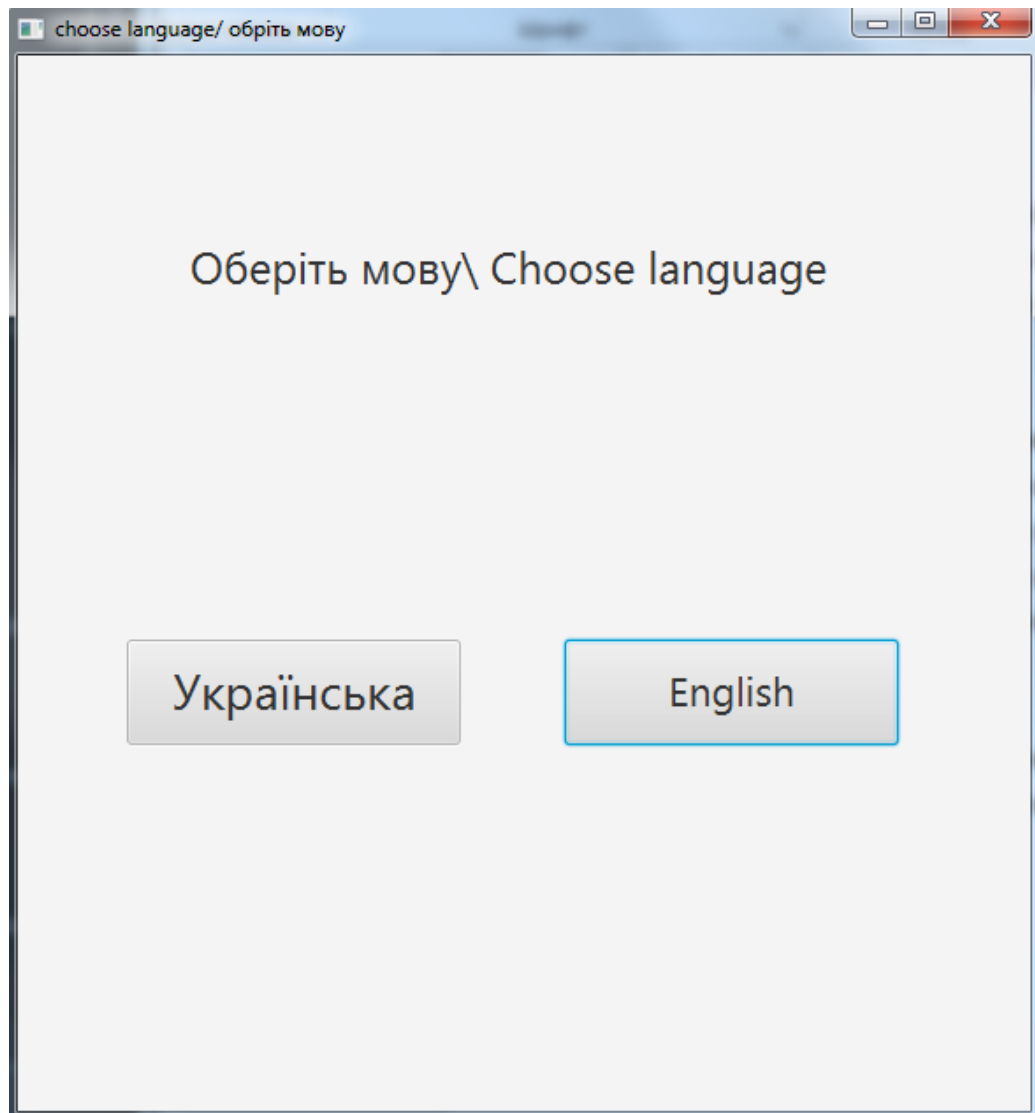


Рисунок 4.3 – Вікно вибору мови інтерфейсу

Перед нами з'являється головне вікно програми (рис. 4.5). Для початку було проведено тестування лекційної частини. Для цього треба натиснути на кнопку «Лекційний матеріал».

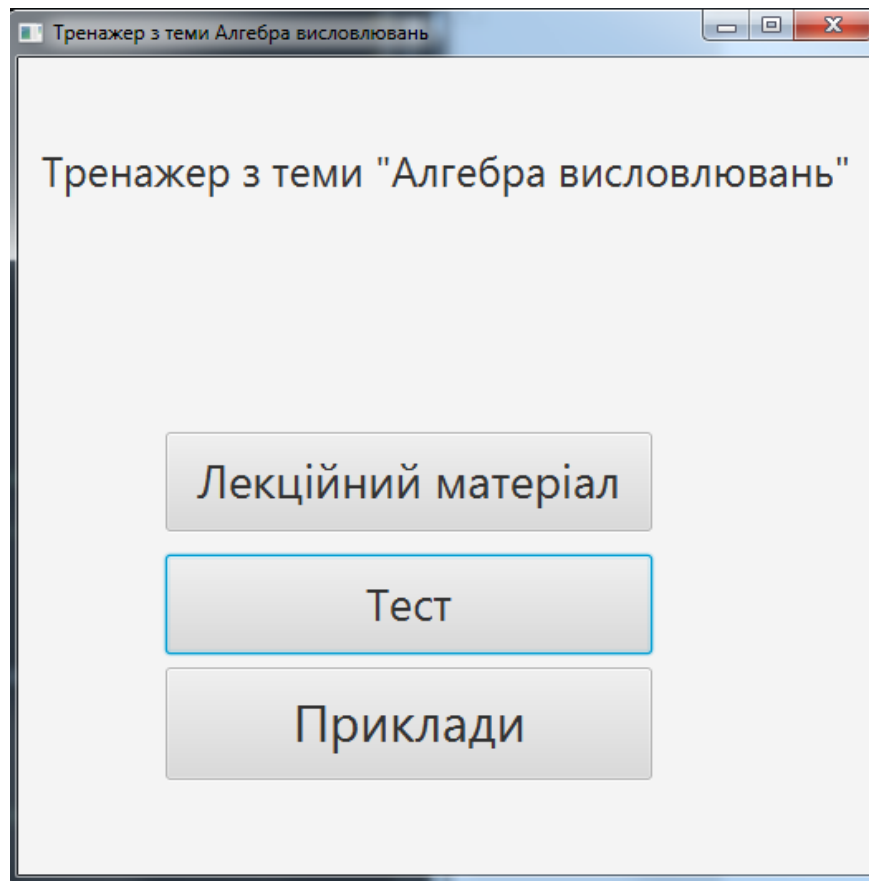


Рис. 4.4 – Головне вікно

У вікні лекційного матеріалу (рис. 2), можна побачити навчальний матеріал та кнопки навігації, вони повинні здійснювати перехід між сторінками лекції (усього їх тринадцять), та кнопка «В головне меню», яка повернути користувача в початковий екран. Також на першому вікні (рис. 2) не повинно бути кнопки «Попередня сторінка» та останньому вікні (рис. 3) не повинно бути кнопки «Наступна сторінка», а в той час на всіх інших вікнах (рис. 4) повинні бути присутні всі описані вище кнопки.

Поняття висловлювання

Висловлювання – це розповідне речення, про яке можна сказати, що воно істинне чи хибне.

Позначаються висловлювання великими латинськими літерами:

$A, B, C, X, Y, Z, A_1, A_2, A_3, \dots$

Приклад:

A = «Київ – столиця України» – істинне висловлювання,

B = «Яблуко – овоч» – хибне висловлювання,

C = «Лимон або овоч, або фрукт» – не є висловлюванням, бо не можна стверджувати ні про істинність, ні про хибність даного речення,

D = «манна каша – смачна страва» – не є висловлюванням, бо поняття «смачна» відносне,

E = «Яка година?» – не є висловлюванням, бо дане речення не є розповідним.

На множині всіх висловлювань визначається функція істинності $\lambda(A)$:

$$\lambda(A) = \begin{cases} 1, \text{ якщо } A - \text{істинне}; \\ 0, \text{ якщо } A - \text{хибне}. \end{cases}$$

Значення $\lambda(A)$ називається **логічним значенням** або **значенням істинності** висловлювання A .

Для наведених вище прикладів $\lambda(A) = 1$, $\lambda(B) = 0$.

[Повернутись в Головне вікно](#)

Сторінка 1

[Наступна сторінка](#)

Рис. 4.2 – Перша сторінка лекції.

Тотожно істинні, тотожно хибні, еквівалентні висловлювання

Приклад:

X = «Не може бути, щоб Матроскін виграв приз і відмовився від нього»;

Y = «Матроскін або не виграв приз, або не відмовився від нього».

Доведемо рівносильність X і Y .

A = «Матроскін виграв приз»;

B = «Матроскін відмовився від приза».

$X = \overline{A \wedge B}$; $Y = \overline{A} \vee \overline{B}$.

A	B	\overline{A}	\overline{B}	$A \wedge B$	$\overline{A \wedge B}$	$\overline{A} \vee \overline{B}$	$X \Leftrightarrow Y$
0	0	1	1	0	1	1	1
0	1	1	0	0	1	1	1
1	0	0	1	0	1	1	1
1	1	0	0	1	0	0	1

[Попередня сторінка](#)

Сторінка 13

[В головне меню](#)

Рис. 4.3 – Остання сторінка лекції

Тренажер з теми Алгебра висловлювань

Логічні операції

Над висловлюваннями визначають наступні логічні операції: заперечення, кон'юнкція, диз'юнкція, імплікація, еквівалентність.

Заперечення (інверсія) висловлювання A – це нове висловлювання, яке істинне, коли A – хибне, і хибне, коли A – істинне.

Заперечення утворюється за допомогою частки «не» або «невірно, що» і позначається \bar{A} (зустрічаються також позначення $\neg A$).

Взаємозв'язок між логічним значенням висловлювання A і логічним значенням його заперечення \bar{A} визначається із наступної таблиці, яку називають *таблицею істинності заперечення*:

$\lambda(A)$	$\lambda(\bar{A})$
0	1
1	0

Приклад:

A = «Петро – відмінник»,
 \bar{A} = «Невірно, що Петро – відмінник»,
 $\bar{\bar{A}}$ = «Петро не є відмінником».

Попередня сторінка
Сторінка 2
Наступна сторінка

В головне меню

Рис. 4.4 – Друга сторінка лекції

Щоб протестувати частину, яка містить тестові запитання, потрібно натиснути на кнопку «Тест». Після натискання повинно з'явитися вікно, де пропонується обрати один з двох режимів «З підказками» та «На кількість балів» (рис. 4.5). Натискання на кнопку «З підказками» повинно з'явитися перше питання (рис. 4.6) та варіанти відповіді на нього. Потрібно обрати відповідь та натиснути на кнопку «Наступне завдання». Якщо відповідь дано правильно, то користувач потрапляє до наступного питання, а в разі неправильної відповіді повинна з'явитися підказка в залежності від того яка відповідь була обрана. Кожному неправильному варіанту відповіді відповідає своя підказка (рис. 4.7 – 4.9), а в разі правильної відповіді користувач потрапляє на наступне запитання.

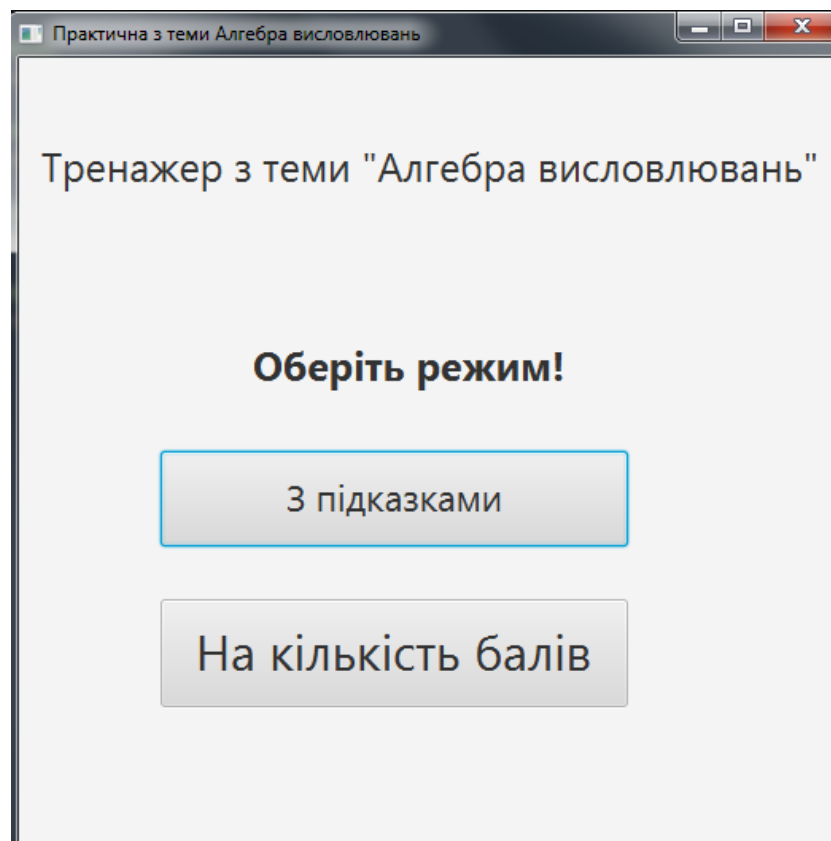


Рис. 4.5 – Вікно вибору режиму

Тестування інших аналогічно першому. Користувач послідовно відповідає на двадцять запитань які розміщені в програмі, шляхом обирання правильної відповіді із запропонованих. В разі неправильної відповіді на будь-яке запитання користувач повинен отримати певну підказку. В залежності від питання та обраної відповіді, повинна відповідати своя підказка яка повинна допомогти користувачу отримати правильний варіант відповіді. У разі неправильної відповіді користувачу надається друга спроба, і так до тих пір поки користувач не дасть правильну відповідь. Після правильної відповіді на усі запитання користувач повертається до головного меню.

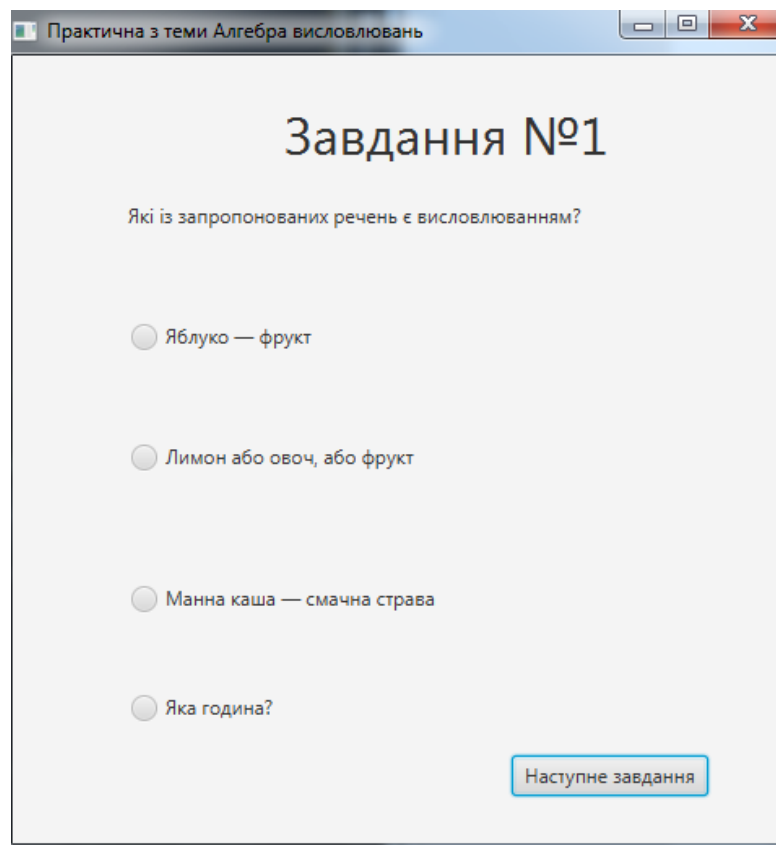


Рис. 4.6 – Перше запитання

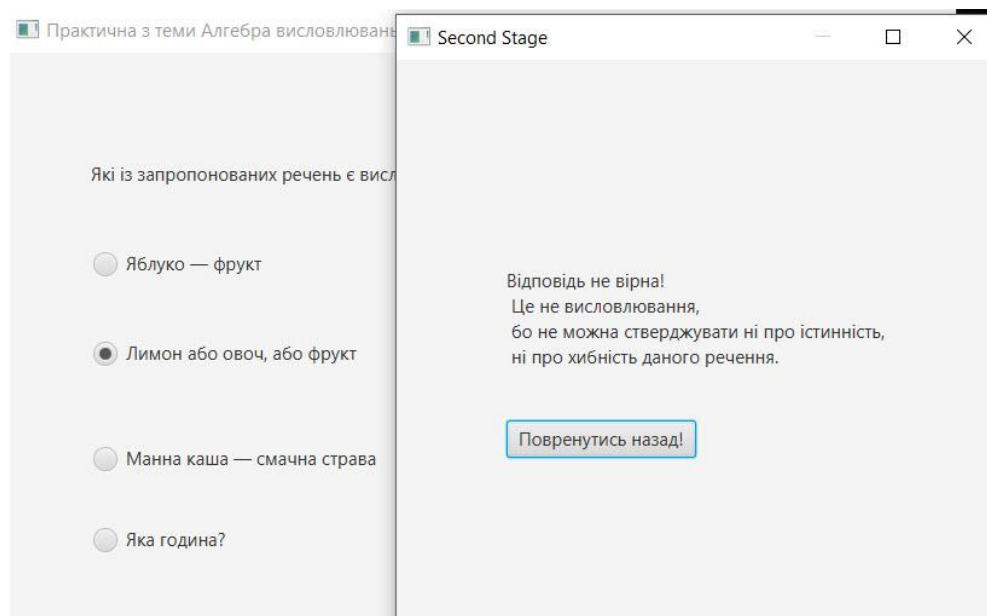


Рис. 4.7 – Перша неправильна відповідь

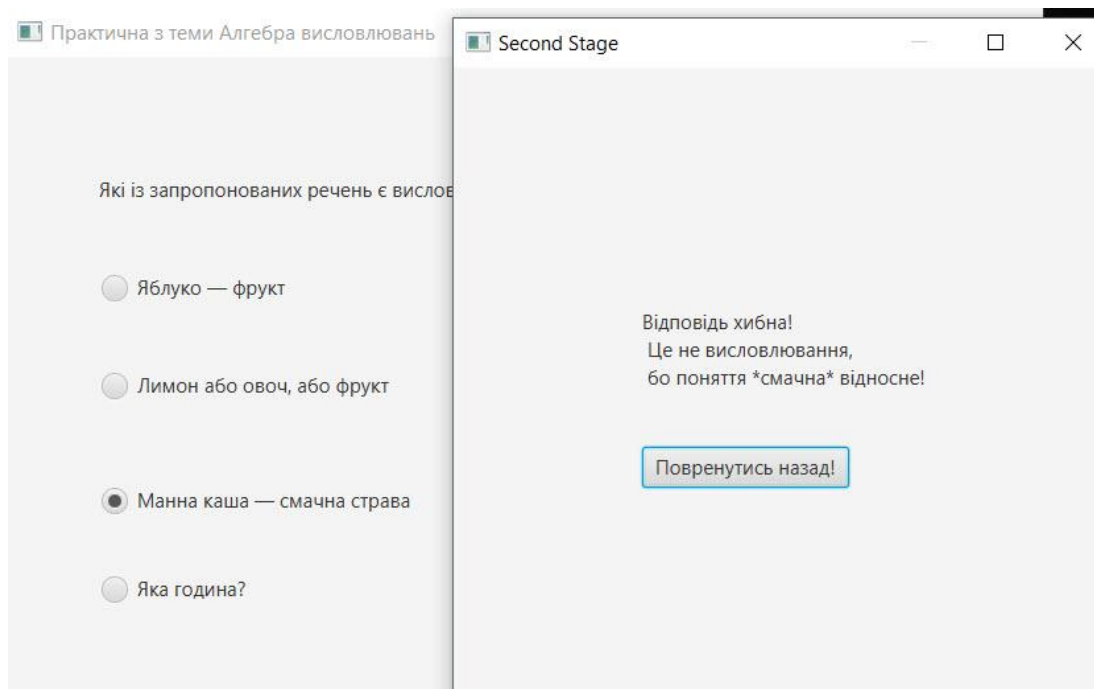


Рис. 4.8 – Друга неправильна відповідь

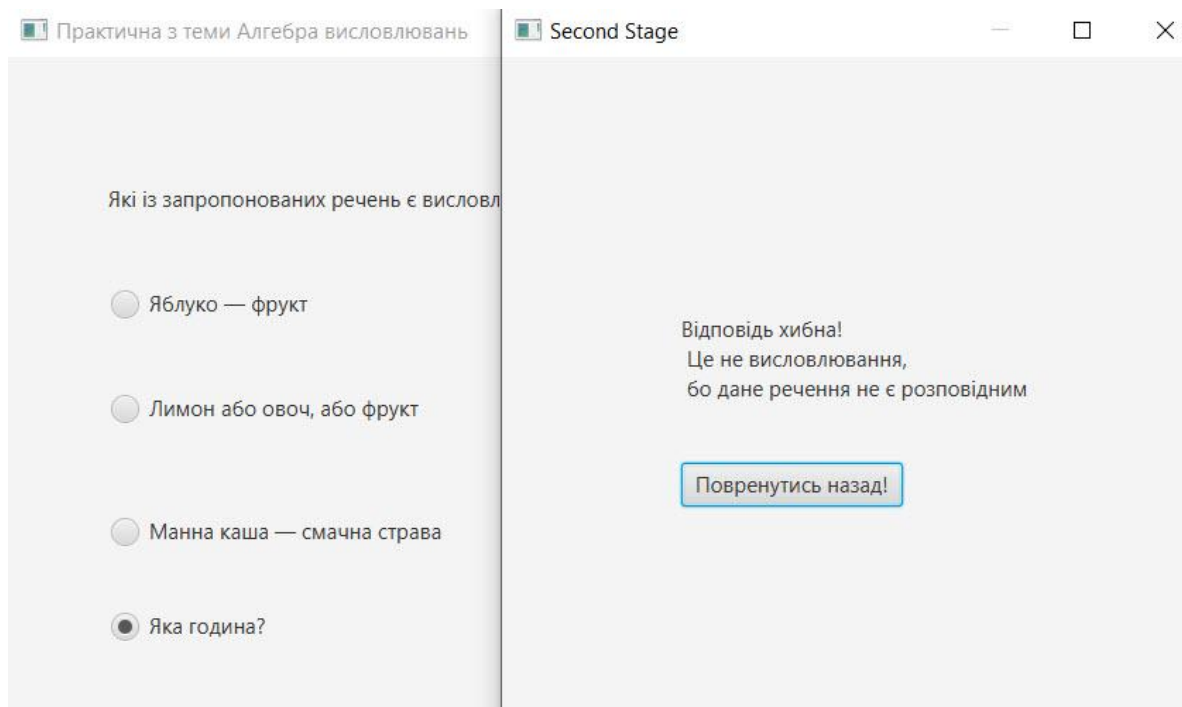


Рис. 4.9 – Третя неправильна відповідь

Для того щоб протестувати другий режим потрібно натиснути кнопку «На кількість балів». Повинно з'явитися вікно з запитаннями та відповідями.

Тестування аналогічне й першому режиму, але в разі неправильної відповіді з'являється вікно не з'являється, а користувач переходить до іншого питання. Під час відповіді на запитання невідомо чи правильна дана відповідь, чи ні. Тому перевірка буде здійснюватися по кількості балів, які ми отримаємо після відповіді на всі запитання (Рис. 4.10). Спочатку тестуємо яка буде оцінка, якщо всі відповіді дано правильно (повинна бути оцінка «5»), яка буде оцінка якщо всі відповіді дано неправильно (повинна бути оцінка «1»), та якщо тільки на половину відповідей оцінка була дана правильно (повинна бути оцінка «3»).

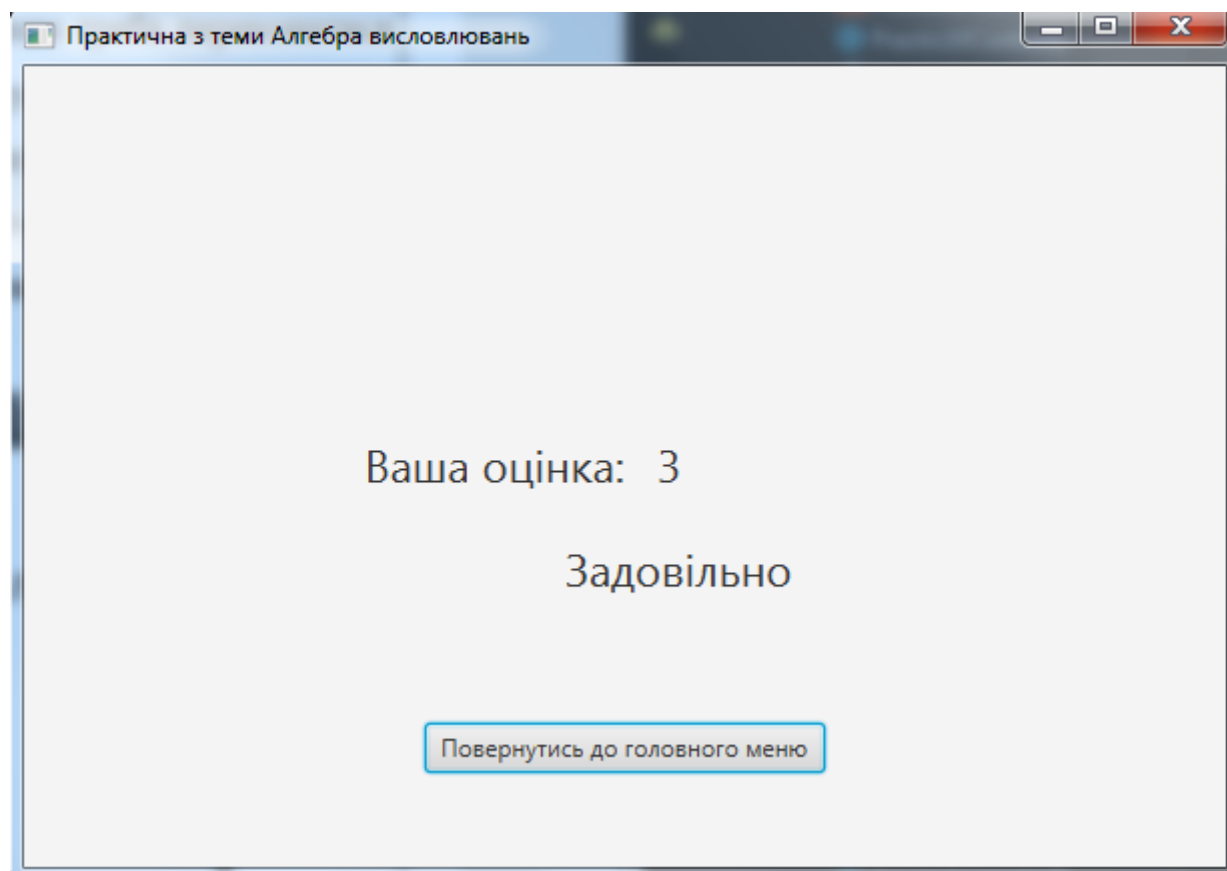


Рис. 4.10 – Вікно з оцінкою за тестування

Для перевірки практичної частини, в якій розв'язуються приклади, потрібно натиснути на кнопку «Приклади». Перед користувачем повинно з'явитися вікно з вибором завдань (рис. 4.11). Користувач може обрати одне з чотирьох завдань для вирішення. Розглянемо одне із завдань.

Спочатку перед користувачем з'являється умова завдання та кнопка «Розпочати» (рис. 4.10). Після натискання на кнопку з'являється перший крок

задачі (рис.4.11). На цьому кроці користувачу потрібно обрати скільки ітерацій має висловлення. Якщо відповідь дано вірно, то користувач переходить другого кроку, а якщо ні, то з'являється повідомлення про помилку (рис. 4.12).

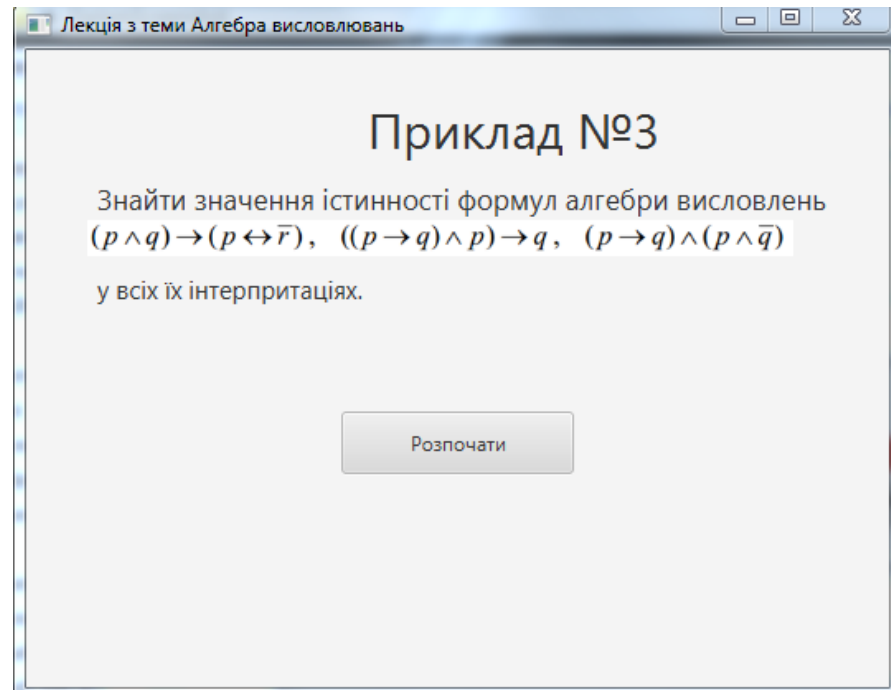


Рисунок 4.10 – Початок завдання.

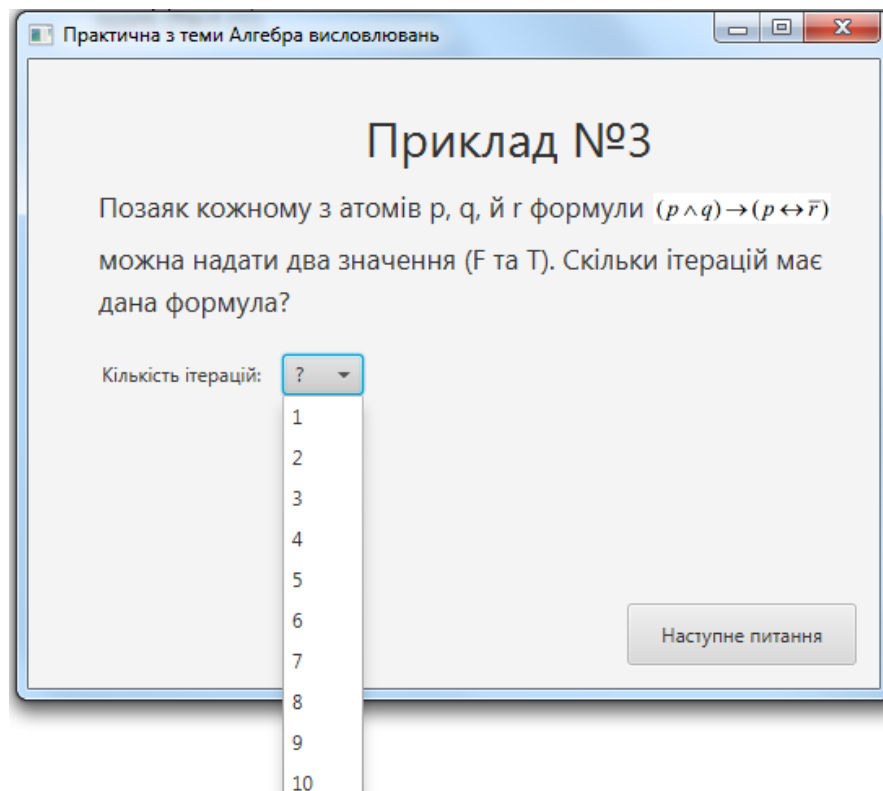


Рисунок 4.11 – Перший крок

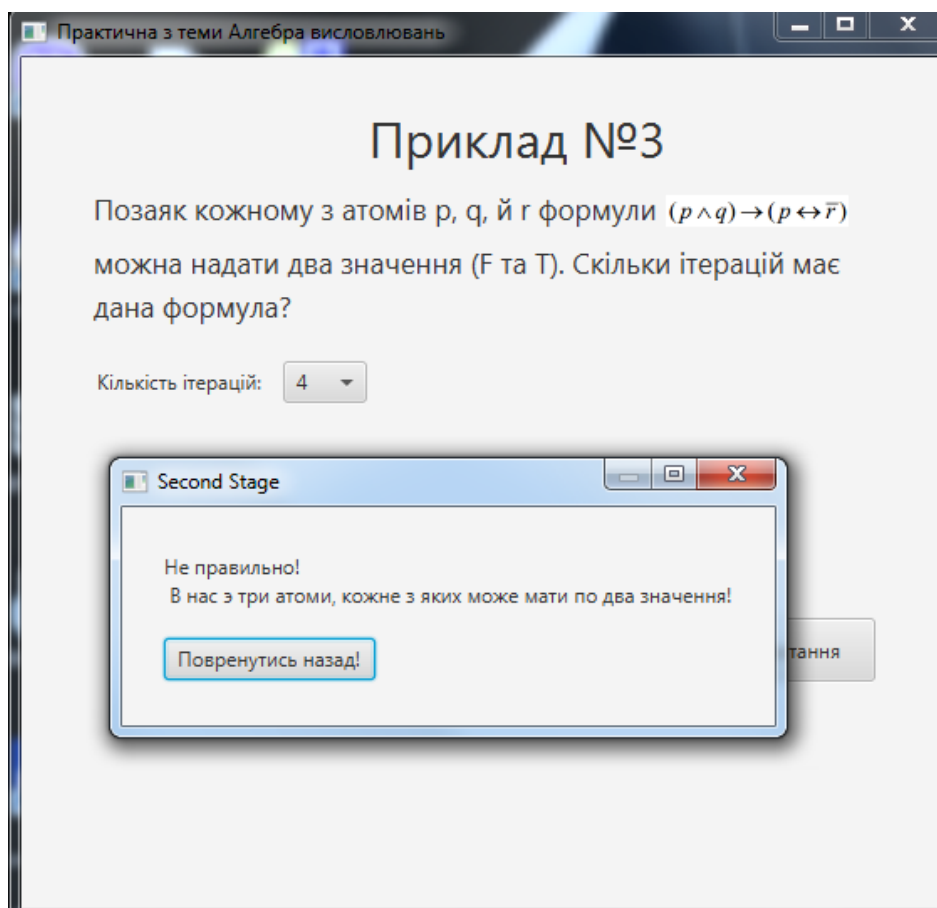


Рисунок 4.12 – Реакція першого кроку на неправильну відповідь.

На другому кроці користувачу потрібно заповнити таблицю (рис 4.13). Потрібно в кожній комірці виставити значення Т або F. Якщо комірки розставлено правильно то користувач потрапляє до наступного кроку, де потрібно заповнити наступну колонку, а в разі помилки з'являється повідомлення про помилку. Аналогічно перевіряємо усі вікна де потрібно заповнити таблицю(рис. 4.14 – 4.16).

Після завершення, йде друга частина завдання. Аналогічно першій частині вказуємо кількість ітерацій та заповнюємо таблицю(рис. 4.17-4.21). Після заповнення таблиці з'являється запитання про висловлення на яке потрібно відповісти (рис. 4.22). Якщо відповісти правильно потрапимо до третьої частини прикладу, де аналогічним способом відповідаємо на запитання та заповнюємо таблицю(рис. 4.23-4.27).

Практична з теми Алгебра висловлювань

Приклад №3

Знайдемо ліву частину виразу $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$

В таблиці виставіть значення колонки $(p \wedge q)$

p	q	r	\bar{r}	$(p \wedge q)$
T	T	T	F	? ▾
T	T	F	T	? ▾
T	F	T	F	? ▾
T	F	F	T	? ▾
F	T	T	F	? ▾
F	T	F	T	? ▾
F	F	T	F	? ▾
F	F	F	T	? ▾

Наступне питання

Рисунок 4.13 – Другий крок

Практична з теми Алгебра висловлювань

Приклад №3

Знайдемо ліву частину виразу $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$

В таблиці виставіть значення колонки $(p \wedge q)$

p	q	r	\bar{r}	$(p \wedge q)$
T	T	T	F	T ▾
T	T	F	T	T ▾
T	F	T	F	F ▾
T	F	F	T	F ▾
F	T	T	F	F ▾
F	T	F	T	F ▾
F	F	T	F	F ▾
F	F	F	T	F ▾

Наступне питання

Рисунок 4.14 – Третій крок

Практична з теми Алгебра висловлювань

Приклад №3

Знайдемо праву половину рівняння $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$

В таблиці виставіть значення колонки $(p \leftrightarrow \bar{r})$

p	q	r	\bar{r}	$(p \wedge q)$	$(p \leftrightarrow \bar{r})$
T	T	T	F	T	<input type="button" value="T"/>
T	T	F	T	T	<input type="button" value="T"/>
T	F	T	F	F	<input type="button" value="F"/>
T	F	F	T	F	<input type="button" value="F"/>
F	T	T	F	F	<input type="button" value="F"/>
F	T	F	T	F	<input type="button" value="F"/>
F	F	T	F	F	<input type="button" value="T"/>
F	F	F	T	F	<input type="button" value="T"/>

Рисунок 4.15 – Четвертий крок

choose language/ обрiть мову

Приклад №3

Знайдемо значення ітерацій формули: $(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$

p	q	r	\bar{r}	$(p \wedge q)$	$(p \leftrightarrow \bar{r})$	$(p \wedge q) \rightarrow (p \leftrightarrow \bar{r})$
T	T	T	F	T	F	<input type="button" value="T"/>
T	T	F	T	T	T	<input type="button" value="T"/>
T	F	T	F	F	F	<input type="button" value="T"/>
T	F	F	T	F	T	<input type="button" value="F"/>
F	T	T	F	F	T	<input type="button" value="F"/>
F	T	F	T	F	F	<input type="button" value="T"/>
F	F	T	F	F	T	<input type="button" value="F"/>
F	F	F	T	F	F	<input type="button" value="T"/>

Рисунок 4.16 –П'ятий крок

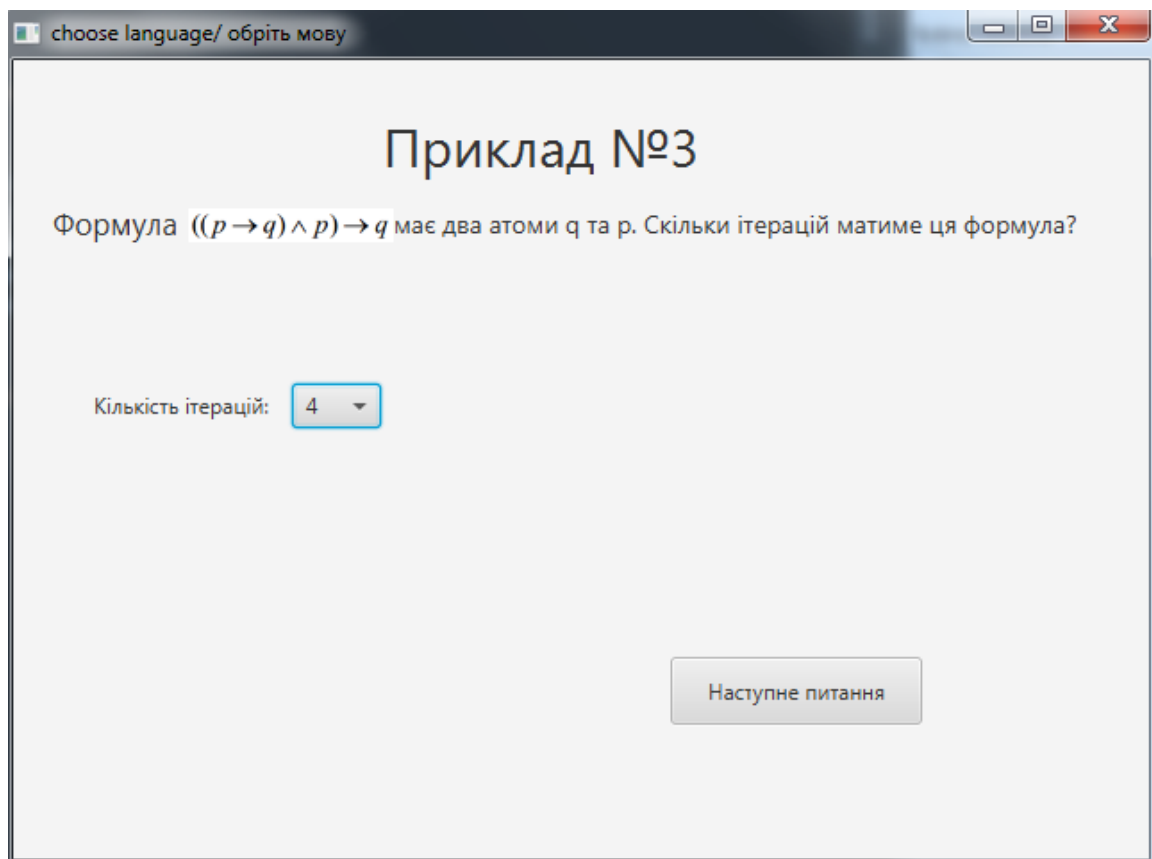


Рисунок 4.17 – Шостий крок

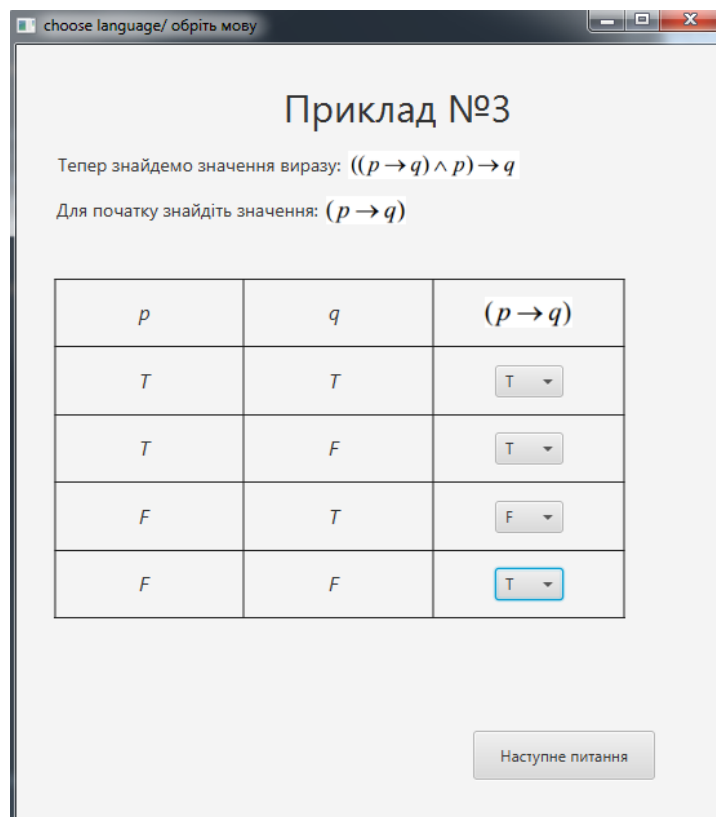


Рисунок 4.18 – Сьомий крок

choose language/ обрiть мову

Приклад №3

Для початку знайдiть значення: $((p \rightarrow q) \wedge p)$

p	q	$(p \rightarrow q)$	$((p \rightarrow q) \wedge p)$
T	T	T	<input type="button" value="T"/>
T	F	F	<input type="button" value="F"/>
F	T	T	<input type="button" value="F"/>
F	F	T	<input type="button" value="F"/>

Рисунок 4.19 – Восьмий крок

Практична з теми Алгебра висловлювань

Приклад №3

Знайдемо значення iстинності формули $((p \rightarrow q) \wedge p) \rightarrow q$

p	q	$(p \rightarrow q)$	$((p \rightarrow q) \wedge p)$	$((p \rightarrow q) \wedge p) \rightarrow q$
T	T	T	T	<input type="button" value="T"/>
T	F	F	F	<input type="button" value="T"/>
F	T	T	F	<input type="button" value="T"/>
F	F	T	F	<input type="button" value="T"/>

Рисунок 4.20 – Дев'ятий крок

Практична з теми Алгебра висловлювань

Приклад №3

Формула $((p \rightarrow q) \wedge p) \rightarrow q$ істинна в усіх чотирьох ітераціях. Як її можна назвати?

☒ Істинна
☐ Компромісна
☐ Тавтологія
☐ Суперечність

Наступне завдання

Рисунок 4.21 – Десятий крок

Практична з теми Алгебра висловлювань

Приклад №3

Формула $(p \rightarrow q) \wedge (p \wedge \bar{q})$ має два атоми q та p . Скільки ітерацій матиме ця формула?

Кількість ітерацій:

Наступне питання

Рисунок 4.21 – Одинадцятий крок

Практична з теми Алгебра висловлювань

Приклад №3

Тепер знайдемо значення виразу: $(p \rightarrow q) \wedge (p \wedge \bar{q})$

Для початку знайдіть значення: \bar{q}

p	q	\bar{q}
T	T	<input type="button" value="F"/>
T	F	<input type="button" value="T"/>
F	T	<input type="button" value="F"/>
F	F	<input type="button" value="T"/>

Рисунок 4.23 – Дванадцятий крок

Практична з теми Алгебра висловлювань

Приклад №3

Тепер знайдемо значення виразу: $(p \rightarrow q)$

p	q	\bar{q}	$(p \rightarrow q)$
T	T	F	<input type="button" value="F"/>
T	F	T	<input type="button" value="T"/>
F	T	F	<input type="button" value="T"/>
F	F	T	<input type="button" value="T"/>

Рисунок 4.24 – Тринадцятий крок

Практична з теми Алгебра висловлювань

Приклад №3

Тепер знайдемо значення виразу: $(p \wedge \bar{q})$

p	q	\bar{q}	$(p \rightarrow q)$	$(p \wedge \bar{q})$
T	T	F	T	<input type="button" value="F"/>
T	F	T	F	<input type="button" value="T"/>
F	T	F	T	<input type="button" value="F"/>
F	F	T	T	<input type="button" value="F"/>

Рисунок 4.25 – Чотирнадцятий крок

Практична з теми Алгебра висловлювань

Приклад №3

Тепер знайдемо значення виразу: $(p \rightarrow q) \wedge (p \wedge \bar{q})$

p	q	\bar{q}	$(p \rightarrow q)$	$(p \wedge \bar{q})$	$(p \rightarrow q) \wedge (p \wedge \bar{q})$
T	T	F	T	F	<input type="button" value="F"/>
T	F	T	F	T	<input type="button" value="F"/>
F	T	F	T	F	<input type="button" value="F"/>
F	F	T	T	F	<input type="button" value="F"/>

Рисунок 4.26 – Пятнадцятий крок

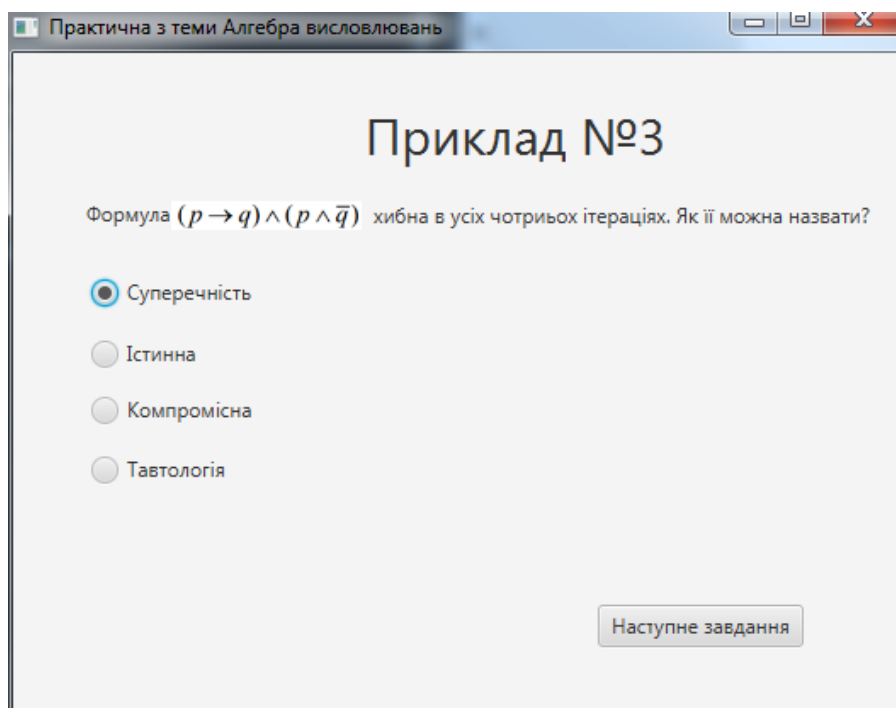


Рисунок 4.27 – Шістнадцятий крок

Після завершення рішення останнього кроку завдання ми завершуємо завдання і потрапляємо до вікна, де можна почати наступне завдання, перейти до списку вибору завдань, перейти до головного меню (рис. 4.28).

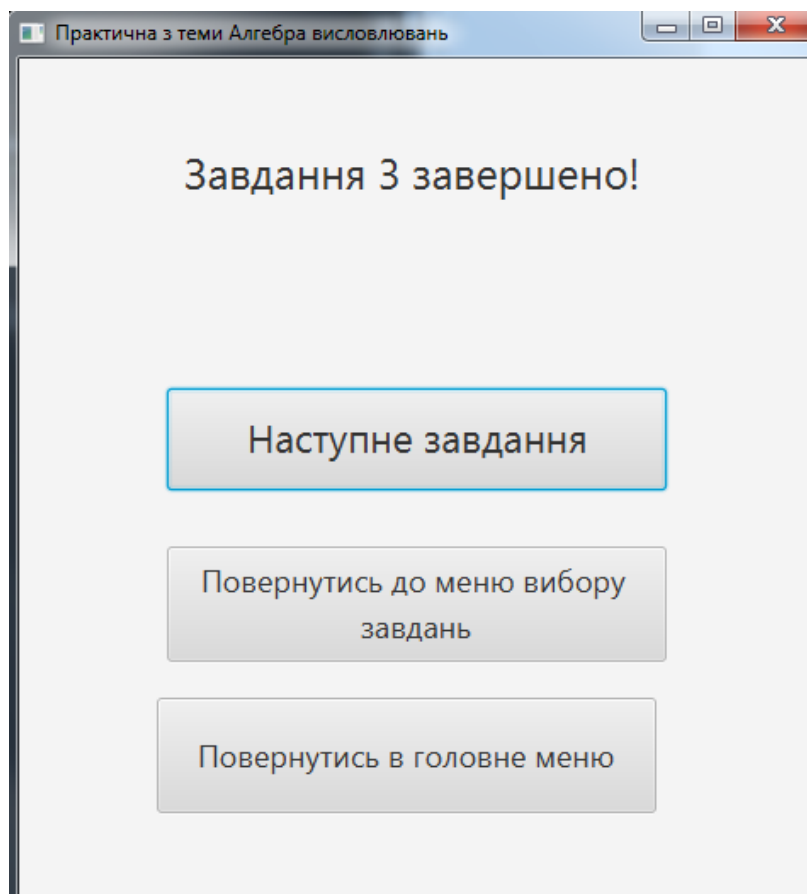


Рисунок 4.28 – Завершення завдання

Тестування інших прикладів проводилося аналогічно. Завдяки тестуванню було підтверджено повну працездатність програми згідно з розробленим алгоритмом.

4.4 Необхідна користувачу програми інструкція

Для початку роботи з програмою необхідно щоб на комп'ютері були встановлені бібліотеки java 8. Потім можна запустити виконуваний файл «Тренажер з теми Алгебра висловлювань.jar». Після запуску з'явиться меню вибору мови програми, де потрібно обрати, на якій мові буде інтерфейс тренажеру(рис. 4.29). Після запуску з'явиться головне вікно програми (рис. 4.30), в якому за допомогою кнопки «Лекційний матеріал» можна перейти до лекції, за допомогою кнопки «Тест», можна перейти до частини, де можна закріпити матеріали прочитані в лекції, а натиснувши кнопку «Приклади» можна потрапити в розділ, де покроковим методом можна розв'язати декілька прикладів з теми «Алгебра висловлювань».

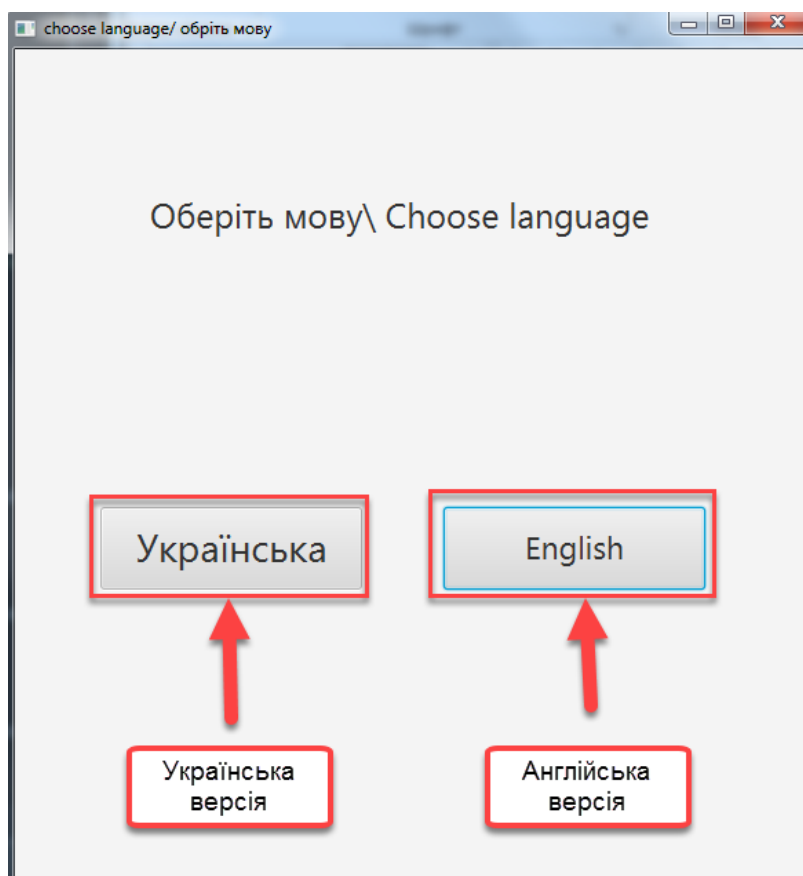


Рисунок 4.29 – Вибір мови тренажера

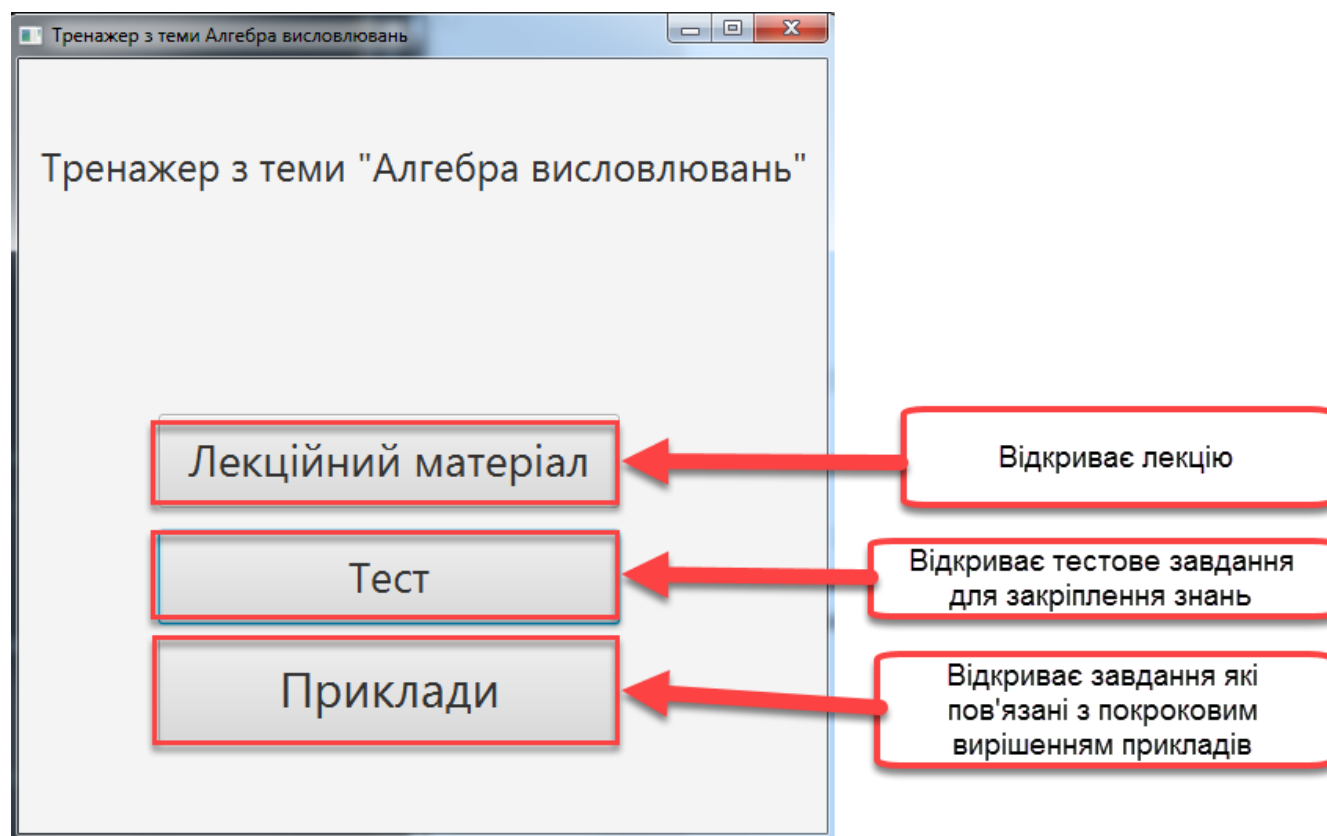


Рис. 4.30 – Головне вікно програми

Після натискання на клавішу «Лекційний матеріал» користувач потрапляє до вікна, в якому можна переглянути лекцію з теми «Алгебра висловлення». Лекція розбита на декілька сторінок (номер сторінки зображений внизу посередині). Перехід між сторінками здійснюється за допомогою кнопок навігації «Наступна сторінка» та «Попередня сторінка». Для повернення до головного меню потрібно натиснути на клавішу «Повернутись в головне меню». Приклад лекційного вікна з описом кнопок зображено на рисунку 4.31.

Тренажер з теми Алгебра висловлювань

Логічні операції

Над висловлюваннями визначають наступні логічні операції: заперечення, кон'юнкція, диз'юнкція, імплікація, еквівалентність.

Заперечення (інверсія) висловлювання A – це нове висловлювання, яке істинне, коли A – хибне, і хибне, коли A – істинне.

Заперечення утворюється за допомогою частки «не» або «невірно, що» і позначається \bar{A} (зустрічаються також позначення $\neg A$).

Взаємозв'язок між логічним значенням висловлювання A і логічним значенням його заперечення \bar{A} визначається із наступної таблиці, яку називають *таблицею істинності заперечення*:

$\lambda(A)$	$\lambda(\bar{A})$
0	1
1	0

Приклад:
 A = «Петро – відмінник»,
 \bar{A} = «Невірно, що Петро – відмінник»,
 \bar{A} = «Петро не є відмінником».

Попередня сторінка

Сторінка 2

Наступна сторінка

Перехід до попередньої сторінки

В головне меню

Повернення до головного вікна

Перехід до наступної сторінки

Рис. 4.31 – Вікно з зображенням лекції

Після натискання на клавішу «Тест» користувач потрапляє до вікна (рис. 4.32), в якому пропонується обрати режим один з двох режимів: «З підказками» або «На кількість балів». В першому режимі потрібно по черзі відповісти на запитання у тестовій формі. Щоб відповісти потрібно із запропонованих варіантів відповідей обрати правильну та натиснути на клавішу «Наступне запитання». Якщо відповідь дано не вірно, то користувач отримає повідомлення про неправильну відповідь і підказка яка допоможе користувачу обрати правильну відповідь, щоб змінити відповідь, потрібно натиснути на кнопку «Повернутись

назад!» та спробувати відповісти ще раз. Якщо відповідь дана вірно, то користувач перейде до наступного запитання. Приклад вікна з зображенням завдання зображено на рисунку 4.33. А в другому режимі при неправильній відповіді користувач не отримає повідомлень і потрапить до наступного завдання, але не отримає балів. В залежності скільки балів отримає користувач, така й оцінка буде чекати його в кінці тесту (рис. 4.34). Після проходження тестів користувач повернеться до головного меню.

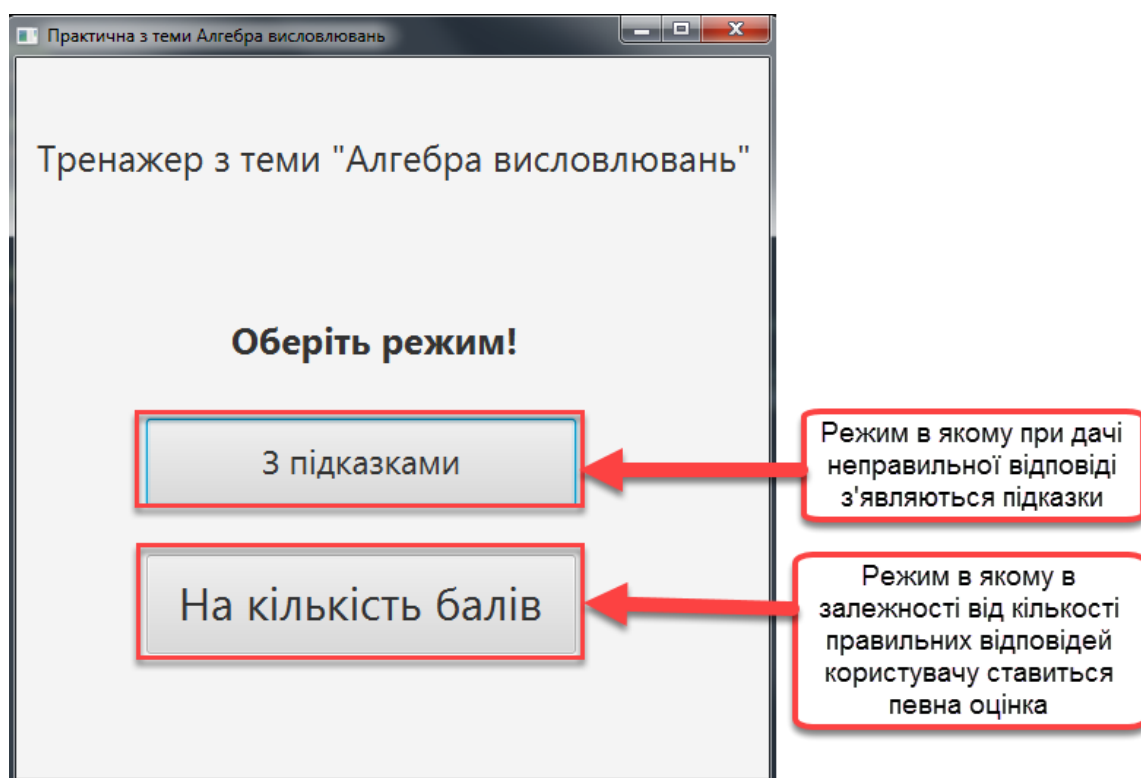


Рисунок 4.32 – Вікно в якому пропонується обрати режим

Після натискання на клавішу «Приклади» з'явиться вікно (рис. 4.35), в якому буде запропоновано обрати приклад для вирішення. Всього в програмі є чотири приклади. Приклади розв'язуються в покроковому режимі. В кожному кроці, свій спосіб вирішення завдання, а саме: обрати одну відповідь (рис. 4.36), обрати декілька відповідей (рис. 4.37), заповнити таблицю (рис. 4.38), поставити певне число (рис. 4.39). Після завершення одного з завдань з'являється вікно (рис. 4.40) за допомогою якого можна почати наступне завдання, перейти до вікна вибору завдань, перейти до головного меню.

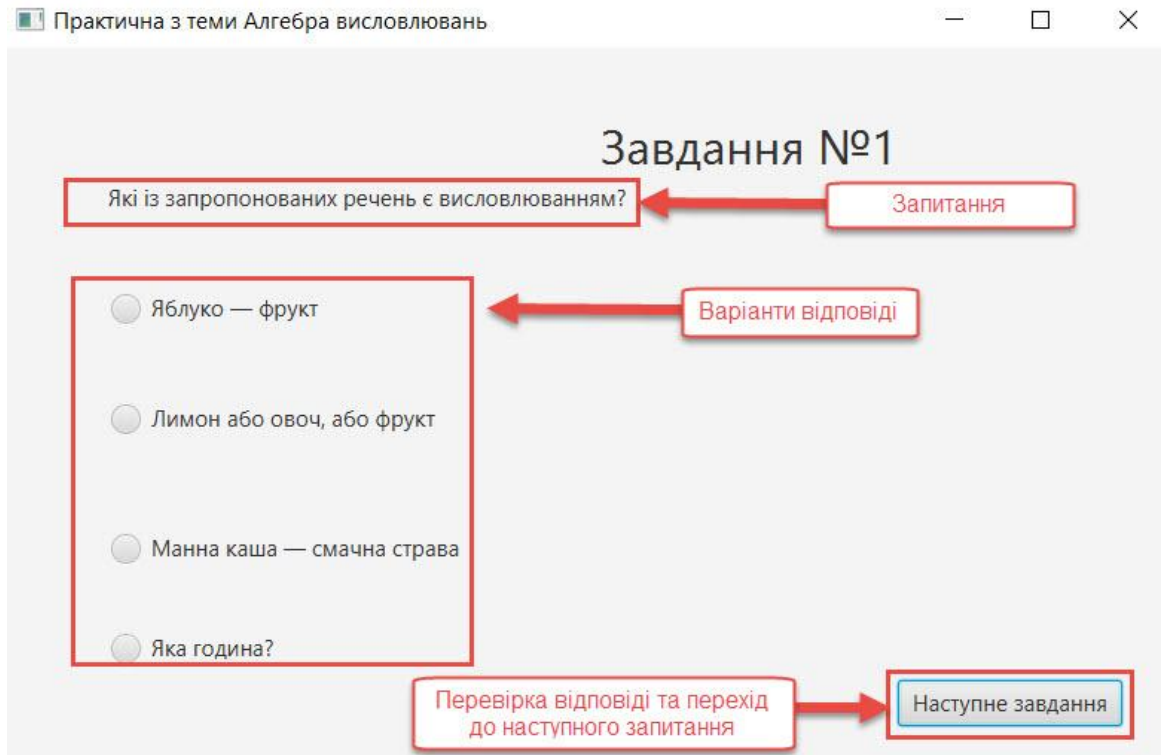


Рис. 4.33 – Вікно з питанням.

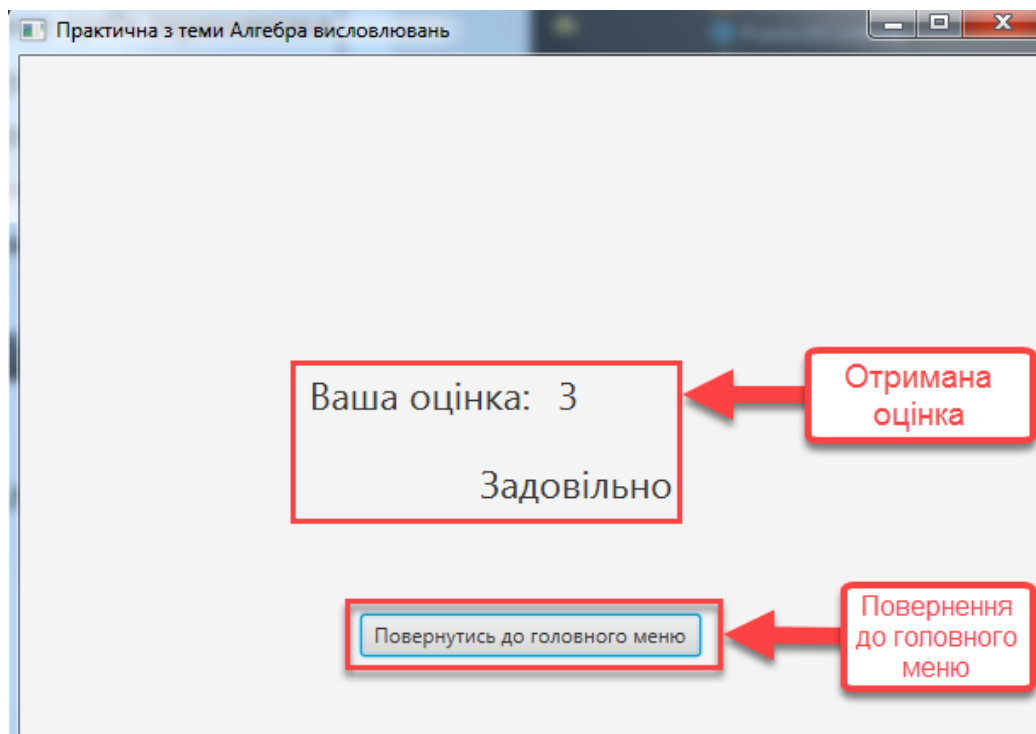


Рис. 4.34 – Вікно з оцінкою

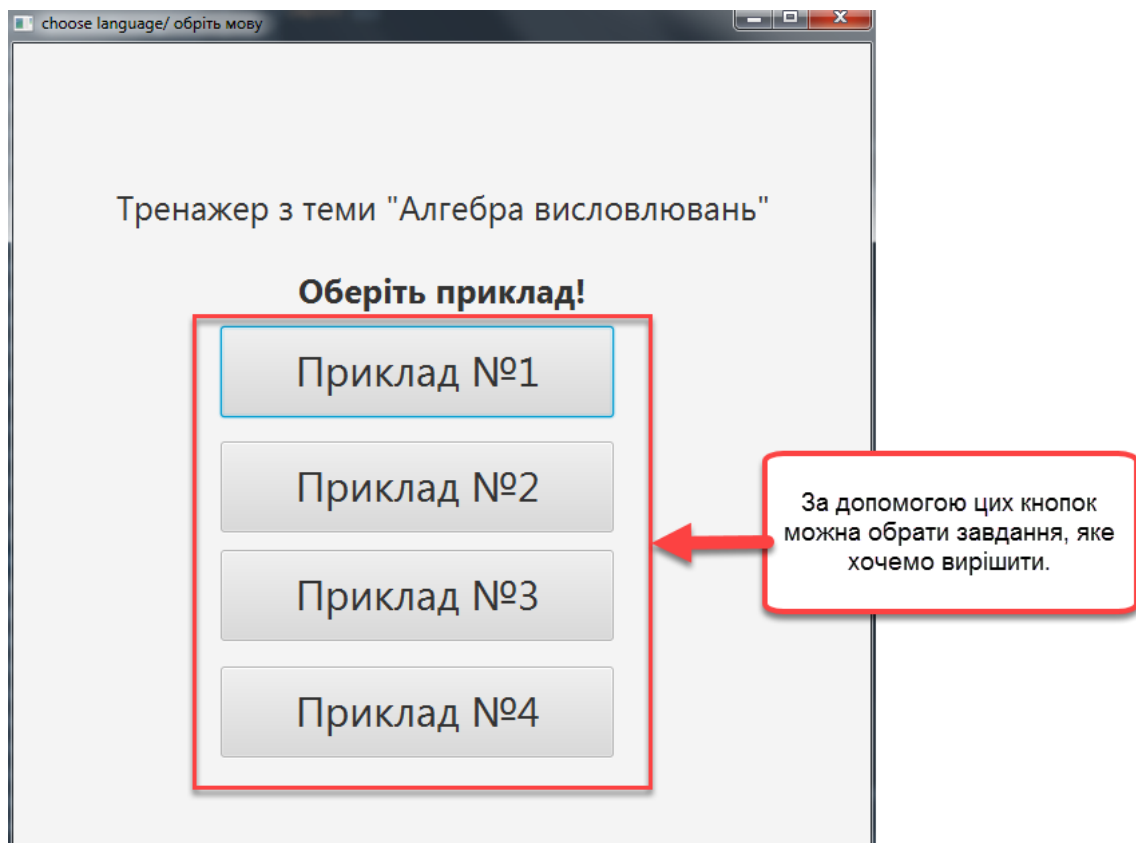


Рисунок 4.35 – Вибір прикладу для розв'язування

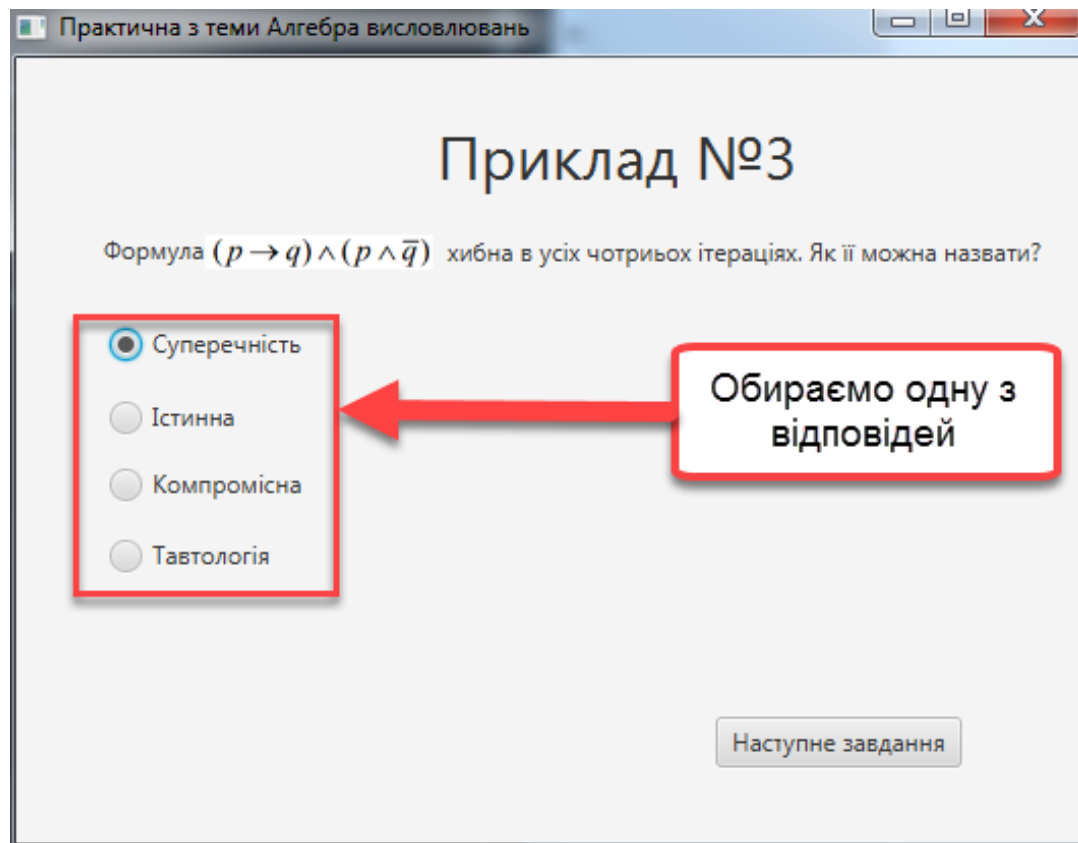


Рисунок 4.36 – Приклад кроку, де потрібно обрати одну відповідь

choose language/ обрiть мову

Приклад №1

Якi з наведених виразiв є висловленням?

<input type="checkbox"/> 1) Кожне дійсне число задовольняє нерівність $x^2 \geq 0$	<input checked="" type="checkbox"/> 8.) Чи правильна велика теорема Ферма?
<input checked="" type="checkbox"/> 2.) Число 168 кратне 9	<input type="checkbox"/> 9.) Рівняння $x^3 + 7x + 1 = 0$ має хоч один дійсний корінь.
<input checked="" type="checkbox"/> 3.) Хай живе математика!	<input checked="" type="checkbox"/> 10.) Розв'язати рівняння $x^2 + 7x + 1 = 0$
<input type="checkbox"/> 4.) 15 кратне 3, але не кратне 4.	<input type="checkbox"/> 11.) Розкрийте підручник на сторінці 23
<input checked="" type="checkbox"/> 5.) Чи існує дійсне число, більше за 3 і менше від $\log_2 9$	<input type="checkbox"/> 12.) Вчитель сказав: "Розкрийте підручник на сторінці 23".
<input checked="" type="checkbox"/> 6.) Ця задача легка.	<input type="checkbox"/> 13.) Якщо $3 < 2$, то $3^2 < 2^2$
<input type="checkbox"/> 7.) Існує найбільше просте число.	

Наступне завдання

Обираємо декілька відповідей

Рисунок 4.37 – Приклад кроку, де потрібно обрати декілька відповідей

choose language/ обрiть мову

Приклад №1

Якi з наведених виразiв є висловленням?

<input type="checkbox"/> 1) Кожне дійсне число задовольняє нерівність $x^2 \geq 0$	<input checked="" type="checkbox"/> 8.) Чи правильна велика теорема Ферма?
<input checked="" type="checkbox"/> 2.) Число 168 кратне 9	<input type="checkbox"/> 9.) Рівняння $x^3 + 7x + 1 = 0$ має хоч один дійсний корінь.
<input checked="" type="checkbox"/> 3.) Хай живе математика!	<input checked="" type="checkbox"/> 10.) Розв'язати рівняння $x^2 + 7x + 1 = 0$
<input type="checkbox"/> 4.) 15 кратне 3, але не кратне 4.	<input type="checkbox"/> 11.) Розкрийте підручник на сторінці 23
<input checked="" type="checkbox"/> 5.) Чи існує дійсне число, більше за 3 і менше від $\log_2 9$	<input type="checkbox"/> 12.) Вчитель сказав: "Розкрийте підручник на сторінці 23".
<input checked="" type="checkbox"/> 6.) Ця задача легка.	<input type="checkbox"/> 13.) Якщо $3 < 2$, то $3^2 < 2^2$
<input type="checkbox"/> 7.) Існує найбільше просте число.	

Наступне завдання

Обираємо декілька відповідей

Рисунок 4.38 – Приклад заповнення таблиці

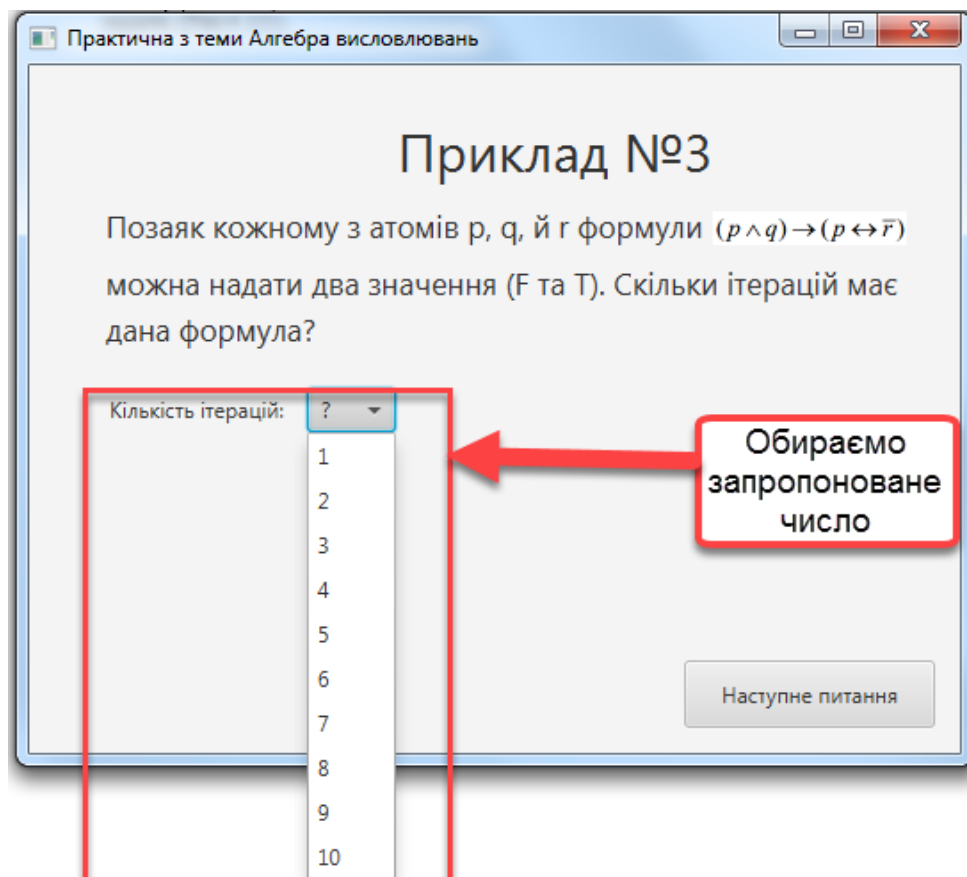


Рисунок 4.39 – Приклад вибору числа із списку запропонованих

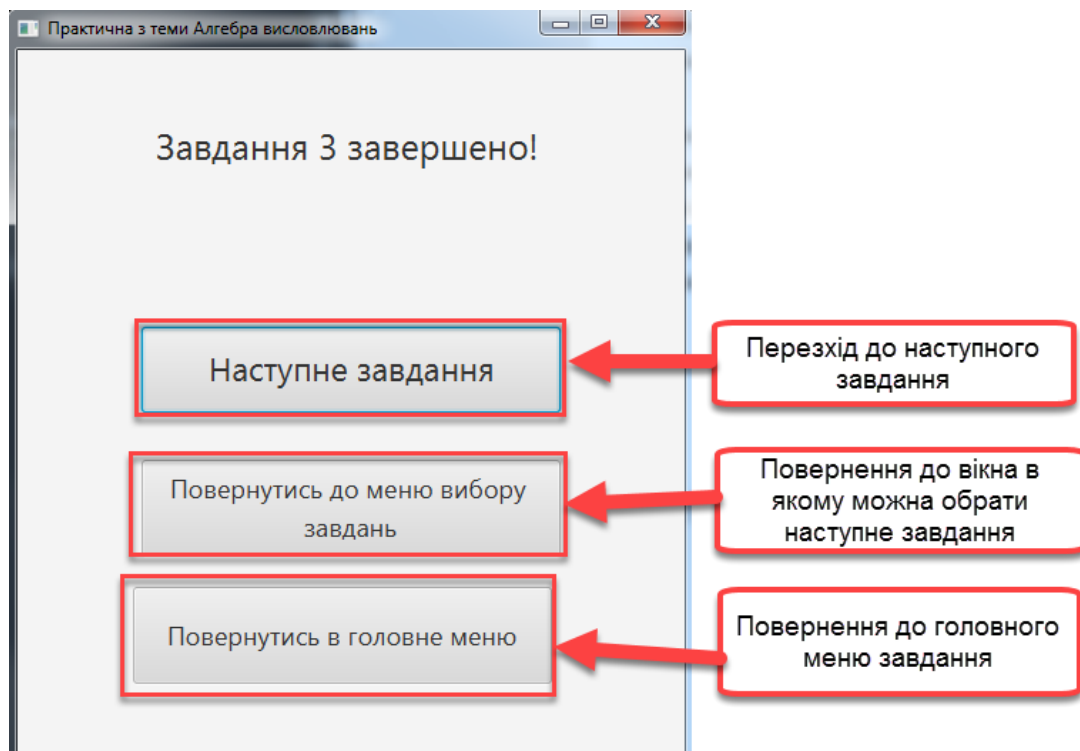


Рисунок 4.40 – Вікно яке сповіщає що завдання завершено.

4.2 Висновки за розділом 4

У результаті виконання практичної частини роботи можна зробити такі висновки:

1. Здійснено опис програмної реалізації та детально описано програмну реалізацію алгоритму описаного в підрозділі 3.2.
2. Здійснено тестування працездатності усіх розділів тренажера. Установлено, що тренажер відповідає розробленому раніше алгоритму, та працює без помилок.
3. Написана інструкція для користувача, в якій описано як працювати з тренажером, та розписано як взаємодіяти з інтерфейсом.

ВИСНОВКИ

За результатами виконання магістерської роботи можна зробити такі висновки:

1. Досліджено стан дистанційного навчання. Проаналізовано його особливості та розвиток. Проаналізовані як переваги, так і недоліки такого виду навчання. Виявлено, що не у всіх учбових закладах є система дистанційного навчання, а там де дистанційне навчання практикується, відсутні тренажери з більшості тем.

2. Аналіз та досліджень джерел та відкритих ресурсів на тренажерів для дистанційного навчання показав, як позитивні так і негативні сторони існуючих на даний момент тренажерів по схожим дисциплінам. Інформації про існування тренажеру з теми «Алгебра висловлення» знайдено не було.

3. Було розглянуто та розв'язано декілька задач з теми « Алгебра висловлювань». На основі розв'язку цих задач було побудовано алгоритм, який можна реалізувати в тренажері. Для спрощення сприйняття тренажеру було розроблено блок-схему, на основі розробленого алгоритму.

4. На основі алгоритму який було побудовано в третьому розділі, було розроблено програмний продукт, що реалізує тренажер з теми «Алгебра висловлювань». Розробка була здійснена за допомогою мови програмування Java. Також в ході тестування було підтверджено повну працездатність тренажера та його відповідність розробленому алгоритму.

5. Тренажер було впроваджено до системи дистанційної системи навчання ПУЕТ, що підтверджує акт про впровадження, та опубліковано наукову статтю, щодо розробленого тренажеру.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Ємець О. О. Методичні рекомендації щодо оформлення пояснювальних записок до дипломних проектів (робіт) для студентів за освітньою програмою «Комп'ютерні науки» спеціальності 122 «Комп'ютерні науки та інформаційні технології», «Комп'ютерні науки» / О. О. Ємець, – Полтава : РВВ ПУЕТ, 2018. – 35 с.
2. Педоренко С.В. Розробка тренажера з теми "М-метод" дистанційного навчального курсу "Методи оптимізації та дослідження операцій" / С.В. Педоренко, О.О. Ємець // Інформатика та системні науки (ІСН-2017): матеріали VIII Всеукраїнської науково-практичної конференції за міжнародною участю, (м. Полтава, 16–18 берез. 2017 р.). – Полтава: ПУЕТ, 2017. – С. 213–216. – Режим доступу: <http://dspace.puet.edu.ua/handle/123456789/5423>
3. Ставковий М.Ю. Розробка тренажеру з теми "Метод аналізу ієрархій" для дистанційного навчання / М.Ю. Ставковий // Інформатика та системні науки (ІСН-2014) : матеріали V Всеукр.наук.-практ. конф., (м. Полтава, 13–15 березня 2014 р.). – Полтава: ПУЕТ, 2014. – С. 296-298. – Режим доступу: <http://dspace.puet.edu.ua/handle/123456789/2843>
4. Крикля М.П. Розробка алгоритму тренажеру з теми "Графічний метод розв'язування задач лінійного програмування"/ М.П. Крикля // Інформатика та системні науки (ІСН-2014) : матеріали V Всеукр.наук.-практ. конф., (м. Полтава, 13–15 березня 2014 р.). – Полтава: ПУЕТ, 2014. – С. 163-165. – Режим доступу: <http://dspace.puet.edu.ua/handle/123456789/2835>
5. Сокол О.В. Розробка тренажера з теми "Нормальні алгоритми" дистанційного навчального курсу «Теорія алгоритмів»/ О.В. Сокол, О.О. Черненко // Інформатика та системні науки (ІСН-2017) : матеріали VIII Всеукр.наук.-практ. конф., (м. Полтава, 16–18 березня 2017 р.). – Полтава: ПУЕТ, 2017. – С. 247-252. – Режим доступу: <http://dspace.puet.edu.ua/handle/123456789/6857>
6. Данник О.І. Розробка алгоритму тренажеру з теми "Графічний метод розв'язування задач лінійного програмування"/ О.І. Данник, О.О.

Черненко//Комп'ютерні науки і прикладна математика (КНіПМ-2018) – Полтава: ПУЕТ, 2018. – С. 247-252. – Режим доступу: <http://dspace.puet.edu.ua/handle/123456789/5481>

7. Бондаренко М.Ф. Комп'ютерна дискретна математика / М. Ф. Бондаренко, Н. В. Білоус, А. Г. Руткас. – Харків.: Компанія СМІТ, 2008. – 480 с.
8. Стрелковська І.В. Дискретна математика: навчальний посібник / І.В. Стрелковська, А.Г. Буслаєв, О.М. Харсун, Т.Л. Пашкова, М.І. Баранов, Т.І. Григор'єва, В.М. Вишневська, Л.Л. Кольцова – Одеса: ОНАЗ ім. О.С. Попова, 2010. –196с. – Режим доступу: www.dut.edu.ua/uploads/1_373_44193539.pdf
9. Трохимчук Р.М. Дискретна математика у прикладах і задачах / Р.М. Трохимчук, М.С. Нікітченко – Київ: КНУ ім. Тараса Шевченка –248с. – Режим доступу: csc.knu.ua/media/filter_public/89/10/89101127-5400-4d61-9840-7eab32caddab/discrete_mathematics.pdf

ДОДАТОК А

Програмний код

Main.java

```
package sample;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class Main extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception{
        Parent root = FXMLLoader.load(getClass().getResource("sample.fxml"));
        primaryStage.setTitle("Тренажер з теми Алгебра висловлювань");
        primaryStage.setScene(new Scene(root, 600, 600));
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

Main.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>
<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.FlowPane?>

<FlowPane alignment="center" hgap="10" vgap="10"
xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="sample.Main">
    <Label layoutX="5.0" layoutY="34.0" prefHeight="36.0"
prefWidth="501.0" text="Тренажер з теми "Алгебра висловлювань"">
        <font>
            <Font size="25.0" />
        </font>
    </Label>
</FlowPane>
```

Controller.java

```
package sample;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
```

```

import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;

public class Controller {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button PracticButton;

    @FXML
    private Button LectionButton;

    @FXML
    void initialize() {
        assert PracticButton != null : "fx:id=\"PracticButton\" was not injected:
check your FXML file 'sample.fxml'.";
        assert LectionButton != null : "fx:id=\"LectionButton\" was not injected:
check your FXML file 'sample.fxml'.";

        LectionButton.setOnAction(event-> {
            LectionButton.getScene().getWindow().hide();

            FXMLLoader loader = new FXMLLoader();
            loader.setLocation(getClass().getResource("Lection.fxml"));

            try {
                loader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }

            Parent root = loader.getRoot();
            Stage Lection = new Stage();
            Lection.setTitle("Лекція з теми Алгебра висловлювань");
            Lection.setScene(new Scene(root));
            Lection.show();
        });
        PracticButton.setOnAction(event-> {
            PracticButton.getScene().getWindow().hide();

            FXMLLoader loader = new FXMLLoader();
            loader.setLocation(getClass().getResource("Practic.fxml"));

            try {
                loader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }

            Parent root = loader.getRoot();
            Stage Practic = new Stage();
            Practic.setTitle("Практична з теми Алгебра висловлювань");
            Practic.setScene(new Scene(root));
            Practic.show();
        });
    }
}

```

```

    }
}

```

Lecture.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.image.*?>
<?import javafx.scene.web.*?>
<?import javafx.scene.text.*?>
<?import javafx.scene.control.*?>
<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.geometry.Insets?>
<?import javafx.scene.layout.GridPane?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>

<GridPane alignment="center" hgap="10" vgap="10"
xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="sample.LectureController">
    <columnConstraints>
        <ColumnConstraints />
    </columnConstraints>
    <rowConstraints>
        <RowConstraints />
    </rowConstraints>
    <children>
        <AnchorPane fx:id="LecImage" prefHeight="660.0" prefWidth="534.0" style="-
fx-background-color: #ffffff;">
            <children>
                <Label layoutX="77.0" layoutY="24.0" text="Поняття висловлювання">
                    <font>
                        <Font size="30.0" />
                    </font>
                </Label>
                <Button fx:id="MainMenuButton" layoutX="14.0" layoutY="586.0"
mnemonicParsing="false" text="Повернутись в Головне вікно" />
                <ImageView fx:id="Image" fitHeight="477.0" fitWidth="548.0"
layoutY="69.0" pickOnBounds="true" preserveRatio="true">
                    <image>
                        <Image url="@Lecture/1.jpg" />
                    </image>
                </ImageView>
                <Button fx:id="NextButton" layoutX="371.0" layoutY="586.0"
mnemonicParsing="false" text="Наступна сторінка" />
                <Label layoutX="267.0" layoutY="591.0" text="Сторінка 1" />
            </children>
        </AnchorPane>
    </children>
</GridPane>

```

LectureController.java

```

package sample;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;
import javafx.fxml.FXML;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;

```

```

import javafx.scene.control.Button;
import javafx.scene.layout.AnchorPane;
import javafx.stage.Stage;
import javafx.scene.image.Image;
import javafx.scene.image.ImageView;

public class LectionController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private AnchorPane LecImage;

    @FXML
    private Button MainMenuButton;

    @FXML
    private Button NextButton;

    @FXML
    void initialize() {
        assert NextButton != null : "fx:id=\"NextButton\" was not injected: check your FXML file 'Lection.fxml'.";
        assert LecImage != null : "fx:id=\"LecImage\" was not injected: check your FXML file 'Lection.fxml'.";
        assert MainMenuButton != null : "fx:id=\"MainMenuButton\" was not injected: check your FXML file 'Lection.fxml'.";
        MainMenuButton.setOnAction(event ->{
            MainMenuButton.getScene().getWindow().hide();

            FXMLLoader loader = new FXMLLoader();
            loader.setLocation(getClass().getResource("sample.fxml"));

            try {
                loader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }

            Parent root = loader.getRoot();
            Stage primaryStage = new Stage();
            primaryStage.setTitle("Тренажер з теми Алгебра висловлювань");
            primaryStage.setScene(new Scene(root));
            primaryStage.show();
        });
        NextButton.setOnAction(event-> {
            NextButton.getScene().getWindow().hide();

            FXMLLoader loader = new FXMLLoader();
            loader.setLocation(getClass().getResource("Lection2.fxml"));

            try {
                loader.load();
            } catch (IOException e) {
                e.printStackTrace();
            }

            Parent root = loader.getRoot();
            Stage primaryStage = new Stage();
            primaryStage.setTitle("Тренажер з теми Алгебра висловлювань");

```

```

        primaryStage.setScene(new Scene(root));
        primaryStage.show();
    });
}
}

```

Practic.fxml

```

<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.text.*?>
<?import java.lang.*?>
<?import java.util.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<AnchorPane prefHeight="484.0" prefWidth="787.0"
xmlns="http://javafx.com/javafx/8" xmlns:fx="http://javafx.com/fxml/1"
fx:controller="sample.PracticController">
    <children>
        <Label layoutX="402.0" layoutY="43.0" text="Завдання №1">
            <font>
                <Font size="32.0" />
            </font>
        </Label>
        <Button fx:id="NextTask" layoutX="602.0" layoutY="424.0"
mnemonicParsing="false" text="Наступне завдання" />
        <Label layoutX="71.0" layoutY="90.0" text="Які із запропонованих речень є
висловлюванням?" />
        <RadioButton fx:id="RB1" layoutX="73.0" layoutY="164.0"
mnemonicParsing="false" text="Яблуко – фрукт" />
        <RadioButton fx:id="RB2" layoutX="73.0" layoutY="238.0"
mnemonicParsing="false" text="Лимон або овоч, або фрукт" />
        <RadioButton fx:id="RB3" layoutX="73.0" layoutY="325.0"
mnemonicParsing="false" text="Манна каша – смачна страва" />
        <RadioButton fx:id="RB4" layoutX="73.0" layoutY="392.0"
mnemonicParsing="false" text="Яка година?" />
    </children>
</AnchorPane>

```

PracticController.java

```

package sample;

import java.io.IOException;
import java.net.URL;
import java.util.ResourceBundle;

import javafx.event.ActionEvent;
import javafx.event.EventHandler;
import javafx.fxml.FXML;
import javafx.geometry.Orientation;
import javafx.geometry.Pos;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.RadioButton;
import javafx.scene.control.ToggleGroup;

```

```

import javafx.scene.layout.FlowPane;
import javafx.scene.layout.StackPane;
import javafx.stage.Modality;
import javafx.stage.Stage;
import javafx.fxml.FXMLLoader;

public class PracticController {

    @FXML
    private ResourceBundle resources;

    @FXML
    private URL location;

    @FXML
    private Button NextTask;

    @FXML
    private Button Error;

    @FXML
    private RadioButton RB1;

    @FXML
    private RadioButton RB2;

    @FXML
    private RadioButton RB3;

    @FXML
    private RadioButton RB4;

    @FXML
    void initialize() {
        assert NextTask != null : "fx:id=\"NextTask\" was not injected: check your FXML file 'Practic.fxml'.";
        assert RB1 != null : "fx:id=\"RB1\" was not injected: check your FXML file 'Practic.fxml'.";
        assert RB2 != null : "fx:id=\"RB2\" was not injected: check your FXML file 'Practic.fxml'.";
        assert RB3 != null : "fx:id=\"RB3\" was not injected: check your FXML file 'Practic.fxml'.";
        assert RB4 != null : "fx:id=\"RB4\" was not injected: check your FXML file 'Practic.fxml'.";

        ToggleGroup group = new ToggleGroup();
        RB1.setToggleGroup(group);
        RB2.setToggleGroup(group);
        RB3.setToggleGroup(group);
        RB4.setToggleGroup(group);

        NextTask.setOnAction(event ->{
            if (RB1.isSelected()){
                {
                    RB1.getScene().getWindow().hide();
                    FXMLLoader loader = new FXMLLoader();

loader.setLocation(getClass().getResource("Practic2.fxml"));

                    try {
                        loader.load();
                    } catch (IOException e) {
                        e.printStackTrace();
                    }
                }
            }
        });
    }
}

```

```

        Parent root = loader.getRoot();
        Stage Practic = new Stage();
        Practic.setTitle("Практична з теми Алгебра висловлювань");
        Practic.setScene(new Scene(root));
        Practic.show();
    }
}
else if (RB2.isSelected()){
    Label lb = new Label("Відповідь не вірна!\n Це не висловлювання,
\n бо не можна стверджувати ні про істинність, \n ні про хибність даного
речення.\n \n ");
    Button bt = new Button ("Повренутись назад!");
    bt.setOnAction(new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent event) {
            bt.getScene().getWindow().hide();
        }
    });
    FlowPane root = new FlowPane(Orientation.VERTICAL, lb, bt);
    root.setAlignment(Pos.CENTER);
    Scene secondScene = new Scene(root, 500, 500 );

    // New window (Stage)
    Stage newWindow = new Stage();
    newWindow.setTitle("Second Stage");
    newWindow.setScene(secondScene);

    // Specifies the modality for new window.
    newWindow.initModality(Modality.APPLICATION_MODAL);
    newWindow.show();
}
else if (RB3.isSelected()){
    Label lb = new Label("Відповідь хибна!\n Це не висловлювання, \n
бо поняття *смачна* відносно!\n \n ");
    Button bt = new Button ("Повренутись назад!");
    bt.setOnAction(new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent event) {
            bt.getScene().getWindow().hide();
        }
    });
    FlowPane root = new FlowPane(Orientation.VERTICAL, lb, bt);
    root.setAlignment(Pos.CENTER);
    Scene secondScene = new Scene(root, 500, 500);

    // New window (Stage)
    Stage newWindow = new Stage();
    newWindow.setTitle("Second Stage");
    newWindow.setScene(secondScene);

    // Specifies the modality for new window.
    newWindow.initModality(Modality.APPLICATION_MODAL);
    newWindow.show();
}
else if (RB4.isSelected()){
    Label lb = new Label("Відповідь хибна!\n Це не висловлювання, \n
бо дане речення не є розповідним\n \n ");
    Button bt = new Button ("Повренутись назад!");
    bt.setOnAction(new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent event) {
            bt.getScene().getWindow().hide();
        }
    });
    FlowPane root = new FlowPane(Orientation.VERTICAL, lb, bt);
    root.setAlignment(Pos.CENTER);

```

```

Scene secondScene = new Scene(root, 500, 500);

// New window (Stage)
Stage newWindow = new Stage();
newWindow.setTitle("Second Stage");
newWindow.setScene(secondScene);

// Specifies the modality for new window.
newWindow.initModality(Modality.APPLICATION_MODAL);
newWindow.show();
}
else {
    Label lb = new Label("Оберіть відповідь!\n \n ");
    Button bt = new Button ("Повренутись назад!");
    bt.setOnAction(new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent event) {
            bt.getScene().getWindow().hide();
        }
    });
    FlowPane root = new FlowPane(Orientation.VERTICAL, lb, bt);
    root.setAlignment(Pos.CENTER);
    Scene secondScene = new Scene(root, 500, 500);

    // New window (Stage)
    Stage newWindow = new Stage();
    newWindow.setTitle("Second Stage");
    newWindow.setScene(secondScene);

    // Specifies the modality for new window.
    newWindow.initModality(Modality.APPLICATION_MODAL);
    newWindow.show();
}

});
}
}

```